

This Page Is Inserted by IFW Operations
and is not a part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

As rescanning documents *will not* correct images,
please do not report the images to the
Image Problems Mailbox.

2

ITO et al.-US Patent Appl. 09/667,230
Ref. F05-494

Citation 1:

Japanese Laid Open Patent Publication H09-171510

Citation 2:

Mobile Media Magazine (7/13/1999) c media Vol.7, No.8, pp 026 - 031, "Cell Phone Information War."

Citation 3:

NIKKEI bytes (5/22/1999) No.191, pp 144 - 149,
"Cell Phone Internet 'Internet at any time and any place' becomes real."

Particularly, Citation 2 describes an EZaccess/EZweb in which "the language specification of HDML includes "a card" comprising a screen display unit and "a deck" comprising a stack of cards. HTML uses "a page" as a screen display unit in which transfer is conducted in page units. However, cell phones have a small sized screen, making transfers in screen units inefficient" (the page 031, left column, lines 1-8).

Claims 1 through 50

The minimum unit could be determined in the invention described in Citation 1 as appropriate based on design preferences. Therefore, the invention described in Claims 1 through 50 of the present application could have been easily invented by one skilled in the Art based on the inventions described in Citations 1 through 3 by applying the invention described in Citation 1 to the systems described in Citations 2 and 3.

3

Record of the Examination Results relating to Documents of the
Prior Art

.

Examined Technical Art IPC 7th Edition G06F12/00

.

Documents of the Prior Art:

NIKKEI communication (3/01/1999), No.289, pp 111 - 117,
"Web: access from anywhere, characters only display in a small
screen"

The record of the examination results relating to documents of
the prior art does not constitute the grounds for rejection.

拒絶理由通知書

特許出願の番号	平成11年	特許願	第271427号
起案日	平成16年	1月19日	
特許庁審査官	廣瀬 文雄	8526	5B00
特許出願人代理人	丸山 隆夫 様		
適用条文	第29条第2項		

この出願は、次の理由によって拒絶をすべきものである。これについて意見があれば、この通知書の発送の日から60日以内に意見書を提出して下さい。

理 由

この出願の請求項1-50に係る発明は、その出願前日本国内又は外国において頒布された下記の刊行物に記載された発明に基いて、その出願前にその発明の属する技術の分野における通常の知識を有する者が容易に発明をすることができたものであるから、特許法第29条第2項の規定により特許を受けることができない。

記

- ref1. 引用例1. 特開平9-171510号公報
- ref2. 引用例2. mobile media magazine、
(1999-7-13)シーメディア Vol.7 No.8、pp.026-031
「携帯電話情報戦争」
- ref3. 引用例3. 日経バイト、(1999-5-22) No.191、pp.144-149
「携帯電話インターネット 「いつでもどこでもすぐに
インターネット」が本番に」

特に、上記引用例2には、EZ access/EZwebに関し「無線転送を効率よくするために、HDMLでは画面表示の単位である「カード」と、それを束ねた「デッキ」が、言語仕様として用意されている。HTMLでは、画面表示の単位は「ページ」そのものであり、そのページ単位で転送が行われるが、画面サイズの小さな携帯電話機で、1画面ごとの転送では効率が悪くなる。」(第031頁左欄第1-8行)と記載されている。

請求項1-50について：

上記引用例1に記載された発明において、最小単位をどのように設定するかは

、必要により適宜行えばよい技術的な設計事項である。したがって、本願請求項 1～50に係る発明は、上記引用例 1 に記載された発明を上記引用例 2 及び 3 に記載されたようなシステムに適用したものであるから、上記引用例 1～3 に記載された発明に基づいて当業者が容易に発明をすることができたものである。

この拒絶理由通知書中で指摘した請求項以外の請求項に係る発明については、現時点では、拒絶の理由を発見しない。拒絶の理由が新たに発見された場合には拒絶の理由が通知される。

先行技術文献調査結果の記録

- ・調査した分野 I P C 第 7 版 G 0 6 F 1 2 / 0 0
- ・先行技術文献 日経コミュニケーション、(1999-3-1) No. 289、pp. 111-117
 ref. 4 「Web編：場所を選ばずアクセス、小さい画面に表示は文字
 だけ」

この先行技術文献調査結果の記録は、拒絶理由を構成するものではない。

9/667230
IDS

ref 1

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平9-171510

(43) 公開日 平成9年(1997)6月30日

(51) Int.Cl. ⁶	識別記号	庁内整理番号	F I	技術表示箇所
G 0 6 F 17/27			G 0 6 F 15/20	5 5 0 E
17/21				5 7 0 R
				5 7 0 D
				5 9 6 B

審査請求 未請求 請求項の数21 F D (全 56 頁)

(21) 出願番号 特願平8-297425

(22) 出願日 平成8年(1996)10月18日

(31) 優先権主張番号 特願平7-296154

(32) 優先日 平7(1995)10月19日

(33) 優先権主張国 日本 (J P)

(71) 出願人 000005496

富士ゼロックス株式会社

東京都港区赤坂二丁目17番22号

(72) 発明者 林 浩一

神奈川県足柄上郡中井町境430 グリーン

テクなかい 富士ゼロックス株式会社内

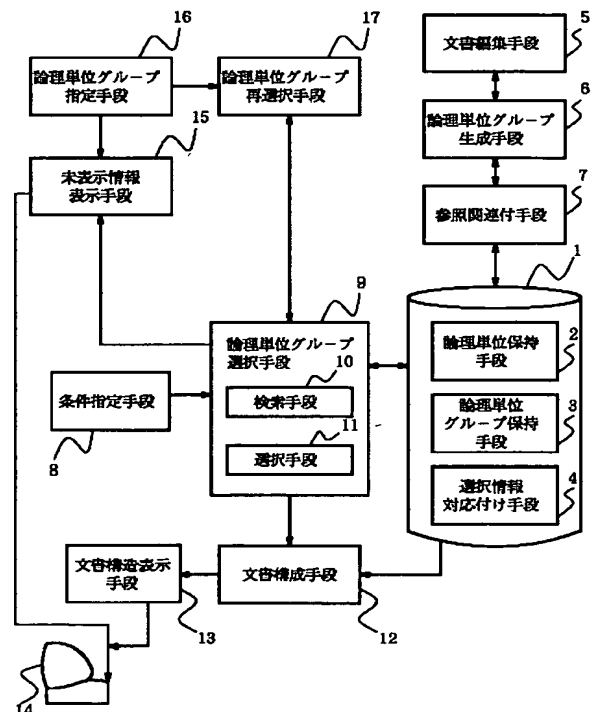
(74) 代理人 弁理士 守山 辰雄

(54) 【発明の名称】 構造化データ処理装置

(57) 【要約】

【課題】 執筆者に編集上の制約を与えずに文書を自由に編集させても、過去の文書の復元や過去には存在していない文書の作成等を可能にして、共同執筆を支障無く実現させる。

【解決手段】 構造化データ処理を構造化文書データに適用した文書処理装置では、論理単位保持手段2に木構造を成している構造化文書の内容の一部分である内容部又は下位階層の構造との関連位置を表すアンカーの少なくともいずれか一方を含む論理単位を保持させ、論理単位グループ保持手段3に構造化文書を構成する木構造の同じ階層に位置する論理単位から成るストリームを保持させる。そして、文書編集手段5によって文書構造を編集すると、論理単位グループ生成手段6が当該文書編集に対応するストリームを新たに生成し、参照関連付手段7が新たに生成されたストリームと所定のアンカーを関連付ける。この結果、新たに生成されたストリームは既存の構造化文書中に組み入れられ、文書編集に応じた構造化文書が作成される。



【特許請求の範囲】

【請求項1】 グラフ構造を成している構造化データのノード情報を保持するノード保持手段と、構造化データのノード間の関連情報を表すリンク情報を保持するリンク保持手段と、

同一のノードを始点とする複数のリンクの内の同一の編集処理に係るリンクを集合として保持するとともに各集合中の含まれるリンク間の位置関係情報を保持するリンク間関係保持手段と、

リンク関係保持手段に保持されたリンク集合中のいずれか1つの集合を特定することによりノード保持手段及びリンク保持手段に保持された情報に基づいて編集対象となる構造化データを特定するカレント構造管理手段と、カレント構造管理手段により特定された構造化データを編集する編集手段と、

編集処理により変更されたリンクを含むリンク集合を生成してリンク間関係保持手段に保持させる登録手段と、を備えたことを特徴とする構造化データ処理装置。

【請求項2】 前記編集手段はノードを追加する処理も行い、また、前記登録手段は追加されたノードをノード保持手段に保持させる処理も行うことを特徴とする請求項1に記載の構造化データ処理装置。

【請求項3】 編集処理に係る情報として選択属性を編集処理により変更されたリンク集合に付与する選択属性対応付け手段を、更に備えたことを特徴とする請求項1又は請求項2に記載の構造化データ処理装置。

【請求項4】 選択属性はバージョン情報であり、選択属性対応付け手段は編集処理に係る構造化データ中の他のリンクに対してもバージョン情報を付与することを特徴とする請求項3に記載の構造化データ処理装置。

【請求項5】 構造化データを再構成させるための選択条件を入力する条件指定手段と、

入力された選択条件に対応する選択属性が付与されているリンク情報をリンク間関係保持手段から選択するリンク集合選択手段と、

選択されたリンク集合に基づいて構造化データを再構成するデータ構成手段と、

を更に備えたことを特徴とする請求項3又は請求項4に記載の構造化データ処理装置。

【請求項6】 リンク集合選択手段は、入力された選択条件に対応するリンク集合が複数存在する場合には、予め設定された選択規則に従っていずれか1つのリンク集合を選択し、

データ構成手段は当該リンク集合に基づいて構造化データを再構成する、ことを特徴とする請求項5に記載の構造化データ処理装置。

【請求項7】 選択規則は特定の利用者情報が付与されているリンク集合を優先的に選択するものである、ことを特徴とする請求項6に記載の構造化データ処理装置。

【請求項8】 選択規則は最新の日時情報が付与されてい

るリンク集合を優先的に選択するものである、ことを特徴とする請求項6に記載の構造化データ処理装置。

【請求項9】 選択規則は特定のバージョン情報が付与されているリンク集合を優先的に選択するものである、ことを特徴とする請求項6に記載の構造化データ処理装置。

【請求項10】 リンク間関係保持手段に保持されている各リンク集合を識別するための情報を表示出力する表示出力手段と、

選択されたリンク集合に基づいてデータ構成手段により再構成された構造化データを表示出力するデータ出力手段と、

を更に備えたことを特徴とする請求項5に記載の構造化データ処理装置。

【請求項11】 選択属性対応付け手段によりリンク集合に選択属性が付与される毎に選択属性の内容を履歴情報として保持する履歴保持手段と、

保持された履歴情報の一覧を表示出力する履歴表示手段と、

を更に備えたことを特徴とする請求項3に記載の構造化データ処理装置。

【請求項12】 木構造を成している構造化文書の内容の一部分である内容単位又は直下の階層の構造との関連位置を表す参照単位の少なくともいずれか一方を論理単位として保持する論理単位保持手段と、

構造化文書を構成する木構造において兄弟の関係にある全ての論理単位から成る論理単位グループを保持する論理単位グループ保持手段と、を備え、

前記参照単位は、該参照単位の直下に位置する少なくとも1つの論理単位グループの集合を表すものであり、更に、

文書構造を編集する文書編集手段と、

該文書編集手段によって文書が編集されたことに基づいて、対応する論理単位グループを新たに生成する論理単位グループ生成手段と、

前記論理単位グループ生成手段により生成された論理単位グループと所定の参照単位を関連付ける参照関連付手段と、

を備えたことを特徴とする文書処理装置。

【請求項13】 前記参照関連付手段は、前記論理単位グループ生成手段によって生成された論理単位グループを、編集前の論理単位グループと同一の集合に追加することを特徴とする請求項12に記載の文書処理装置。

【請求項14】 木構造を成している構造化文書の内容の一部分である内容単位及び直下の階層の構造を表す参照単位を保持する論理単位保持手段と、

構造化文書を構成する木構造において兄弟の関係にある全ての内容単位あるいは参照単位からなる論理単位グループを保持する論理単位グループ保持手段と、を備え、

前記参照単位は、該参照単位の直下に位置する少なくと

10

20

30

40

50

3

も1つの論理単位グループの集合を表すものであり、更に、

前記論理単位グループと該論理単位グループを選択するための選択情報とを対応付ける選択情報対応付け手段と、

文書構造を構成するために前記選択情報を含む選択条件を指定する条件指定手段と、

前記条件指定手段によって指定された選択条件と前記選択情報対応付け手段によって対応付けられている選択情報に基づいて、前記論理単位グループ保持手段に保持されている論理単位グループを選択する論理単位グループ選択手段と、

前記論理単位グループ選択手段により選択された論理単位グループに基づいて文書を構成する文書構成手段と、を備えたことを特徴とする文書処理装置。

【請求項15】 前記選択情報対応付け手段は、選択情報として文書のバージョン情報を前記論理単位グループ保持手段に保持された論理単位グループに対応付けることを特徴とする請求項14に記載の文書処理装置。

【請求項16】 前記選択情報対応付け手段は、選択情報として文書の更新日時情報を前記論理単位グループ保持手段に保持された論理単位グループに対応付けることを特徴とする請求項14に記載の文書処理装置。

【請求項17】 前記選択情報対応付け手段は、選択情報として文書の利用者情報を前記論理単位グループ保持手段に保持された論理単位グループに対応付けることを特徴とする請求項14に記載の文書処理装置。

【請求項18】 前記選択情報対応付け手段により前記論理単位グループに対応付けられる選択情報は、バージョン情報、更新日時情報、利用者情報であり、前記条件指定手段により指定される選択条件は文書のバージョンと日時情報と利用者情報を含むものであり、前記論理単位グループ選択手段は、前記条件指定手段により指定されたバージョン情報に対応する更新日時情報以降で、指定された日時情報以前の更新日時情報と対応付けられ、且つ、指定された利用者情報に一致する論理単位グループを検索する検索手段と、該検索手段による検索の結果、複数の論理単位グループが存在した場合にいずれか1つの該論理単位グループを選択する選択手段と、を有し、

前記論理単位グループ選択手段により選択された論理単位グループから構成された文書構造を表示する表示手段を有することを特徴とする請求項14に記載の文書処理装置。

【請求項19】 前記選択手段は、前記検索手段による検索の結果、論理単位グループが存在しなかった場合に、指定されたバージョン情報に一致する論理単位グループを選択することを特徴とする請求項18に記載の文書処理装置。

【請求項20】 構造化文書の内容の一部である内容

4

単位又は直下の階層の構造を表す参照単位の中から編集対象を指定するステップと、

指定した内容単位又は参照単位を編集するステップと、該編集に対応して、構造化文書を構成する木構造において兄弟の関係にある全ての内容単位あるいは参照単位から成る論理単位グループを生成するステップと、を含み、

前記参照単位は、該参照単位の直下に位置する一つ以上の論理単位グループの集合を表すものであり、更に、

10 該論理単位グループを選択するための選択情報を入力するステップと、

入力された選択情報と新たに生成された論理単位グループを関連付けるステップと、

該論理単位グループを前記指定した内容単位あるいは参照単位を含む論理単位グループの集合に追加するステップと、を含むことを特徴とする文書処理方法。

【請求項21】 文書の内容の一部である内容単位と直下の階層の構造を表す参照単位を有する構造化文書を選択するための選択条件を入力するステップと、

20 前記構造化文書を構成する木構造において兄弟の関係にある全ての内容単位又は参照単位からなる論理単位グループの中から、入力された選択条件と論理単位グループに対応付けられている選択情報とに基づいて論理単位グループを選択するステップと、を含み、

前記参照単位は、該参照単位の直下に位置する少なくとも一つの論理単位グループの集合を表すものであり、更に、

選択された論理単位グループより文書を構成するステップと、

30 構成された文書を表示するステップと、を含むことを特徴とする文書処理方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】 本発明は、文書データ等と言ったグラフ構造を成す構造化データを作成、編集、保存等する処理装置及び方法に関し、特に、執筆者にデータ編集上の制約を与えずに自由に編集することを許しても、過去の構造化データの復元や共同執筆を支障無く行うことのできる構造化データ処理装置及び方法に関する。

【0002】

【従来の技術】 一般に、ファイル等といった文書情報のまとまりに更新が繰り返された場合に、特定の条件に合致する過去の状態の文書を取り出すことを保証する技術はバージョン管理と呼ばれ、過去の文書の状態はバージョンと呼ばれる。このようなバージョン管理は過去の状態の文書を復元できるという効用に加え、複数の担当者による共同執筆作業を支援できる等の効用がある。

【0003】 ここで、従来では、バージョン管理はファイル等の情報のまとまりを単位にして行われており、フ

ファイルを共同編集するときには、複数の担当者が同じファイルを同時には編集できないように制限する等の管理が行われていた。しかしながら、非常にサイズの大きな文書を執筆する場合や、複数の担当者が文書を執筆する場合には、章や節等といった部分単位で文書を編集する方が便利である。そこで、文書を構造化して部分毎のバージョンを管理して、各担当者が分担を決めて執筆するように取り決めたり、同一の部分複数の担当者で同時に編集できないように制限を加えることによって、複数の担当者による共同執筆が可能にした発明がなされている。

【0004】従来における構造化した文書のバージョン管理を、図84～図89を参照して更に詳しく説明する。文書の構造は順序付きの木構造として表現され、章や節等といった部分単位を表すノードが親子関係を示すリンクを用いて構造化され、この木構造の末端に内容（章や節等といった部分単位の内容）を含んだノードが接続される。すなわち、図84に示す文書構造の例では、ノードAを文書ルートとすると、ノードBは章、ノードC及びDが節に該当し、末端のノード

(u)、(v)、(x)がテキスト等のデータ内容を含む内容部となる。なお、章や節等といった部分単位は文書全体としての構造によって定義されているので、文書の全体構造が変化してしまうとバージョン管理の単位が安定しなくなってしまう。そこで、このような従来における文書構造のバージョン管理では、文書の全体構造を固定し、内容部についてバージョン管理を行っている。

【0005】そして、新たな内容部を作成して既存の内容部と入れ替えるといった編集処理を行う場合には、図85に示すように、新たな内容部(y)を既存の内容部(x)と入れ替えるとともに、古い内容部(x)は元の位置に対応付けて記憶管理する。これによって、当該文書の過去の状態（すなわち、編集前の状態）を復元するときには、古い内容部(x)を新しい内容部(y)と再び入れ替えるだけで、所期の目的を達成することができる。

【0006】このようなバージョン管理では、上記した理由から文書の全体構造を変更した場合には、これを管理することができず、上記のように古い内容部を新しい内容部と入れ替えるだけでは過去のバージョンを復元することはできない。例えば、図85に示した元の文書に対して、図86に示すように新たな部分単位の内容部(z)を追加して対応する新たな部分構造(ノード)Eを文書構造に挿入した場合、図87に示すように内容部分(x)に係る部分構造(ノード)Dを削除した場合、図88に示すように部分構造(ノード)Cと部分構造(ノード)Dとの順序を入れ替えた場合、図89に示すように内容部分(x)に係る部分構造(ノード)Dの位置を移動させた場合等においては、文書の全体構造が変更されるため、内容部を入れ換えるだけでは過去の状態

の文書を復元することはできない。

【0007】なお、特開平6-35914号公報には、図85に示したような内容部のバージョン管理を利用して、同一の文書に対する共同執筆を支援する文書作成システムが記載されている。この文書作成システムでは、複数の執筆担当者によって同じノードの内容部を編集したときに、それぞれの担当者が編集した内容部に対応させてバージョンノードを作成し、それぞれの内容部をバージョンノードとの対応付けで管理する。更に、それぞれのバージョンノードにはアクセス権を設定しておき、執筆担当者が文書構造中の内容部を編集する際には、バージョンノードに基づいて、その担当者に適合するアクセス権が設定されている内容部を提示してその編集を可能にしている。

【0008】また、特開平7-44563号公報や特開平6-131343号公報には、文書の構造をノードとリンクからなる木構造で表現し、文書内の互いに内容が異なる複数の部分について論理構造関係が同一の場合には、ノードを多重化することによりこれら部分の論理構造を複合して1つの複合文書構造として保持し、更に、それぞれの部分構造を区別するための属性を多重化したノードに付与しておくことにより、複数の木構造の中から必要な文書構造に対応する木構造を選択することができる多重文書処理システムが記載されている。この多重文書処理システムによれば、文書構造を全体構造と内容に単純に分割するよりも柔軟に表現することができ、更に、必要に応じて複数の木構造の中から必要な文書構造に対応する木構造を選択することができる。

【0009】ここで、バージョン管理技術は、文書データの編集以外にも、ソフトウェア開発、CADデータベースなど様々な分野で利用されている技術であり、一般のバージョン管理技術においても、それぞれの部品（すなわち、部分データ）がバージョンを持った複雑なデータの全体構造についてのバージョン管理は重要な課題である。

このようなバージョン管理に関する従来技術としては、例えば、下記のような種々な技術が知られている。

【0010】複雑な構造を持つCADデータベースのバージョン管理技術について網羅的に記述した“R. Katz, 'Toward a Unified Framework for Version Modeling in Engineering Databases', ACM Computing Surveys, vol. 22, no. 4, 1990”では、上述した全体構造のバージョンをコンフィギュレーションと呼んで、部品のバージョンとは別に管理する技術が示されている。この文献の中には様々なバージョン管理技術が示されているが、いずれもバージョン管理の目的は過去の状態を復元することである。そのための方法として、導出木を用いる方法とレイヤを用いる方法が示している。導出木を用いる方法は、変更の履歴としてどのバージョンのものからどのバージョンが派生したかという関係で導出木(derivation

tree)を管理し、その中から一つの状態を選択することによって、過去の状態を復元する。このように導出木を利用して、一つ前の状態からの差分だけを記録することによって、バージョン管理の実現に必要な記憶容量を軽減させることができる。なお、"Sten Minor & Boris Magnusson, 'A Model for Semi-(a)Synchronous Collaborative Editing, proceedings of ECSCW '93, pp. 219-231, 1993"では、導出木による全体構造のバージョン管理を文書編集に応用した例を示している。

【0011】レイヤによる方法では、変更を行ったときに、生成された部品や更新された部品をレイヤというまとまりとして保持し、適切にレイヤを重ね合わせることで過去に存在していた状態を表現できるようにする。重ね合わされた複数のレイヤ中に同じ部品のバージョンが含まれているときには、後から重ねたレイヤの部品を選択することによって、必要な時点での全体構造の状態を構成する部品のバージョンが選ばれる。なお、"Goldstein, 'Layered Networks as a Tool for Software Development', proceedings of 7th International Conference on Artificial Intelligence, pp. 913-919, 1981"では、レイヤを用いた方法をソフトウェアのバージョン管理に利用した例を示している。

【0012】また、オブジェクト指向データベースのバージョン管理について記述した"Cellary, 'Consistency of Versions in Object-Oriented Databases', proceedings of International VLDB Conference, pp. 432-441, 1990"では、特定の全体構造に属する部品にバージョン名を対応づけることによって、指定したバージョンの全体構造の復元を容易にする手法を示している。

また、ハイパーテキストのバージョン管理について記述した"Osterbye, 'Structural and Cognitive Problems in Providing Version Control for Hypertext', Proceedings of ACM Conference on Hypertext (ECHT'92), pp. 33-42, 1992."では、ハイパーテキストに含まれる個々のノードのバージョンから、作成日時などの条件で適切なものを選択することによって、コンフィギュレーションを構成する手法を示している。

【0013】

【発明が解決しようとする課題】上記のように、従来のバージョン管理にあつては、文書の全体構造を固定化しておけば過去の文書のバージョンを復元することが可能であるが、実際の文書の編集においては、図86～図89に示したように文書の全体構造の変更も内容部の変更に伴って行われる場合が珍しくはない。また、実際の文書執筆の過程においては、下記に示す例(1)～(3)のように、編集がなされた文書に対して、編集前の文書を復元させる必要が生ずる場合が多々ある。

【0014】(1) 特定の日時における文書の取り出し
例えば、既に削除してしまった過去に書いた文書を取り出すために、日時を指定して、過去の時点で編集してい

た文書のバージョンを復元させる。

(2) 特定の目的で使した文書の取り出し
例えば、中間報告等、これまでの執筆過程で一時的にまとめて配布した文書のバージョンを復元させる。

(3) 特定の人に更新された文書の取り出し
例えば、複数の担当者で執筆がなされているときに内容をチェックするために、特定の人による更新が行われる前のバージョンや、行われた後のバージョンを復元させる。

10 【0015】また、過去に存在したバージョンを復元することの必要性の他に、過去に存在しなかった文書の構造を取り出すことが必要になることがある。一般に仕事かどのように進行するかは予め決められないことが多く、進行に伴って予測していなかった様々な要求が現れるからである。そのため、あらかじめ担当者とマージの手順をきちんと決めて文書の執筆を進めても、途中でうまくゆかなることが多い。要求を満たすためには、それまでに作ってきた文書を当初想定していなかった状況に合わせて再構成する必要が生ずる。例えば、複数人で一つの調査報告書を書いているときに、現在執筆中の最新報告書の章立てではなく過去に別の目的で作った報告書の章立てに、他のメンバーの書いた最新の情報を取り入れて文書を作る必要が生じた場合、取り出されなければならない文書構造は、過去に存在したものではなく新しい目的のために過去の部品から合成したものになる。

20 【0016】しかしながら、従来のバージョン管理にあつては、文書の全体構造を固定化しておくことを前提として過去の文書の復元が可能となるものであるため、上記のような実際の文書処理作業に即した復元処理を行うことができなかった。なお、内容部及び文書の全体構造の変更を自由に行つた場合であっても、文書の全体構造と内容部との両方のバージョンを指定するようにすれば、過去の文書構造を復元を実現することが可能であるが、このためには、利用者は文書の全体構造の変更過程の全容を把握しておかなければならないため、利用者に非常に困難且つ煩雑な作業を強いることになって実用的ではない。

30 【0017】更に言えば、実際の文書執筆の過程では、自分の担当している内容部の変更の影響が他の内容部の変更につながることも多いので、自分の担当範囲以外の内容部についても一貫して変更できる方が望ましい。すなわち、他の担当者によって編集集中の内容部を同時に編集することや、他の担当者に断りなく文書の全体構造を変更することも、実際の文書執筆の過程では必要となる。このように各担当者が自由に編集等を行つた場合であっても、各担当者毎の過去の文書を復元できれば支障はないが、従来のバージョン管理にあつては、文書の全体構造を固定化しておくことを前提として過去の文書の復元が可能となるものであるため、各担当者
50 は、常に編集上の制約を守らなければならず、執筆作業

を煩わしいものにしていた。

【0018】なお、上記した多重文書処理システム（特開平7-44563号公報、特開平6-131343号公報）にあっては、ノードに付与された属性を指定することにより担当者毎の過去の文書を復元することも可能であるが、いずれのノードをどのように多重化し、また、これらのノードをどのような区別の仕方をするかについては、全て利用者に任されているため、利用者は極めて煩雑な操作を強いられることとなる。更に言えば、共同執筆や或る時点でのバージョンの復元に当該多重文書処理システムを利用することを考えると、複数の担当者が作成した部分的に内容が異なる複数の文書を管理する能力を持っているが、実際に共同執筆や必要なバージョンの復元を可能にするためには、ノードの多重化のタイミングやそのときに与える属性について煩雑な取り決めが必要になる。したがって、実際の利用に際しては極めて煩雑な操作や知識が必要とされ、当該多重文書処理システムは実質的にはバージョンの復元や共同執筆を支援するための手段とは言えなかった。

【0019】また、従来のバージョン管理技術は、本質的に過去に存在していた構造の中から必要なものを選択する技術であるため、利用者の指示にしたがって当初想定していなかった新しい状況に応じた適切な構造を取り出すことはできなかった。例えば、導出木を用いる方法では、取り出せる構造は導出木のノードとして表される過去の状態だけであり、また、“Cellary”の方法では、過去に存在したバージョンに名前を付けるので過去に存在した構造しか取り出せない。

また、レイヤによる方法は、レイヤの順序を変えることで過去になかった構造も作り出せるが、取り出したい構造が取り出せるか、また、取り出すためにどのようにレイヤの順序を決めればよいのかを知ることができない。したがって、新しい状況に対応するための適切な構造を利用者が指示して取り出すことはできない。また、“Osterbye”の方法も、適切な条件を与えれば必要な構造を取り出せる可能性を示しているが、実際にどのような条件を与えればよいのかは示していない。したがって、新しい状況に対応するための適切な構造を利用者が指示して取り出すことはできない。

【0020】本発明は、上記従来の事情に鑑みなされたもので、文書データ等のグラフ構造をなす構造化データについて、利用者に自由に編集をさせても、過去のデータ構造の復元や新しい状況に合わせた適切なデータ構造の作成を可能ならしめることを目的とする。更に具体的に言えば、本発明は、執筆者に編集上の制約を与えずに文書を自由に編集させても、過去の文書の復元や共同執筆を支障無く実現させることを目的とする。また、本発明は、文書の全体構造が変更された場合にあって、必要な過去の文書バージョンを支障無く復元できる構造化文書を構成することを目的とする。また、本発明は、利

用者による簡易な指定操作によって、適切な過去の文書バージョンを復元させることを目的とする。また、本発明は、利用者による簡易な指定操作によって、過去に作ってきた文書を元にして、当初想定していなかった新しい状況に合わせた適切な文書を作り出すことを目的とする。

【0021】

【課題を解決するための手段】本発明の構造化データ処理装置では、ノード保持手段にグラフ構造を成している構造化データのノード情報を保持し、リンク保持手段に構造化データのノード間の関連情報を表すリンク情報を保持し、リンク間関係保持手段に同一のノードを始点とする複数のリンクの内の同一の編集処理に係るリンクを集合として保持するとともに各集合中の含まれるリンク間の位置関係情報を保持している。そして、編集対象となる構造化データは、リンク関係保持手段に保持されたリンク集合中のいずれか1つの集合を特定することにより、ノード保持手段及びリンク保持手段に保持された情報に基づいて特定されてカレント構造管理手段で管理されており、この特定された構造化データを編集手段により編集処理を行う。そして、編集処理によりリンクに変更が加えられたときには、登録手段が変更されたリンクを含むリンク集合を生成してリンク間関係保持手段に保持させる。

【0022】なお、前記編集手段はノードを追加する処理も行い、また、前記登録手段は追加されたノードをノード保持手段に保持させる処理も行う。また、編集処理に係る情報として選択属性を編集処理により変更されたリンク集合に付与する選択属性対応付け手段を更に備えており、特に選択属性としてバージョン情報を付与するときには、選択属性対応付け手段は編集処理に係る構造化データ中の他のリンクに対してもバージョン情報を付与する。

【0023】また、本発明の構造化データ処理装置では、条件入力手段から構造化データを再構成させるための選択条件を入力すると、リンク集合選択手段が入力された選択条件に対応する選択属性が付与されているリンク情報をリンク間関係保持手段から選択し、データ構成手段が選択されたリンク集合に基づいて構造化データを再構成する。なお、リンク集合選択手段は、入力された選択条件に対応するリンク集合が複数存在する場合には、利用者情報を優先させる、日時情報を優先させる、バージョン情報を優先させる等といった、予め設定された選択規則に従っていずれか1つのリンク集合を選択し、データ構成手段は当該リンク集合に基づいて構造化データを再構成する。

【0024】ここで、本発明を木構造を成している構造化文書データの処理に適用した場合には、本発明は下記のような文書処理装置又は文書処理方法として実現される。なお、以下の記載において、論理単位はノードとリ

ンクとの少なくともいずれか一方を含むデータ構造中の単位（部分データ）に該当する。

【0025】本発明の文書処理装置では、論理単位保持手段に木構造を成している構造化文書の内容の一部分である内容単位又は直下の階層の構造との関連位置を表す参照単位の少なくともいずれか一方を論理単位として保持させ、論理単位グループ保持手段に構造化文書を構成する木構造において兄弟の関係にある全ての論理単位から成る論理単位のグループ（集合）を保持させる。なお、参照単位は、該参照単位の直下に位置する少なくとも1つの論理単位グループの集合を表すものであり、参照単位で示された論理単位グループによって文書全体が構成される。また、兄弟の関係とは、同じ論理単位（親）を直上にもつ論理単位同士の関係をいう。

【0026】更に、本文書処理装置では、文書編集手段によって文書構造を編集すると、論理単位グループ生成手段が当該文書編集に対応する論理単位グループを新たに生成し、参照関連付手段が新たに生成された論理単位グループと所定の参照単位を関連付ける。この結果、新たに生成された論理単位グループは既存の構造化文書中に組み入れられ、文書編集に応じた構造化文書が作成される。また、新たに生成された論理単位グループは、対応する編集前の論理単位グループを同一の集合に追加されて構造化文書中に組み入れられ、これら論理単位グループは同じ階層の集合にまとめられて論理単位グループ保持手段に保持される。

【0027】また、本発明の文書処理装置では、更に、選択情報対応付け手段によって論理単位グループと該論理単位グループを選択するための選択情報とを対応付け、条件指定手段によって文書構造を構成するために前記選択情報を含む選択条件を指定すると、選択情報対応付け手段によって対応付けられている選択情報に基づいて、論理単位グループ選択手段が論理単位グループ保持手段に保持されている論理単位グループを選択する。そして、文書構成手段が論理単位グループ選択手段により選択された論理単位グループに基づいて文書を構成する。

【0028】この選択情報としては、例えば、文書のバージョン情報、文書の更新日時、文書の利用者情報等が用いられ、これら選択情報が選択情報対応付け手段によって論理単位グループ保持手段に保持されている論理単位グループに対応付けられる。したがって、選択情報に基づいて構成された文書は指定された文書のバージョン情報、文書の更新日時、文書の利用者情報等に応じたものであり、過去の文書バージョンがこれら選択情報に基づいて復元される。

【0029】また、本発明の文書処理装置では、更に、選択情報対応付け手段により論理単位グループに対応付けられる選択情報を、バージョン情報、更新日時情報、利用者情報とし、条件指定手段により指定される選択条

件は文書のバージョンと日時情報と利用者情報を含むものとする、そして、論理単位グループ選択手段に備えられた検索手段によって、条件指定手段により指定されたバージョン情報に一致する論理単位グループに対応付けられた更新日時情報以降で、指定された日時情報以前の更新日時情報と対応付けられ、且つ、指定された利用者情報に一致する論理単位グループを検索し、更に、論理単位グループ選択手段に備えられた選択手段によって、検索手段による検索結果で複数の論理単位グループが存在した場合には、いずれか1つの論理単位グループを選択する。そして、この選択された論理単位グループを表示手段が表示し、利用者に対応し且つ指定された日時を基準とした最新のバージョンの文書が検索されて表示される。

【0030】また、本発明の文書処理方法では、構造化文書の内容の一部分である内容単位又は下位階層の構造を表す参照単位の中から編集対象を指定し、この指定した内容単位又は参照単位を編集すると、編集に対応して、構造化文書を構成する木構造において兄弟の関係にある全ての内容単位あるいは参照単位から成る論理単位グループが生成される。そして、新たに生成された論理単位グループを選択するための選択情報が入力されると、入力された選択情報と新たに生成された論理単位グループとを関連付け、該論理単位グループを前記指定した内容単位あるいは参照単位を含む論理単位グループの集合に追加する。したがって、新たに生成された論理単位グループは既存の構造化文書中に組み入れられ、文書編集に応じた構造化文書が作成される。

【0031】また、本発明の文書処理方法では、文書の内容の一部分である内容単位と下位階層の構造を表す参照単位を有する構造化文書を選択するための選択条件を入力すると、構造化文書を構成する木構造において兄弟の関係にある全ての内容単位又は参照単位からなる論理単位グループの中から、入力された選択条件と論理単位グループに対応付けられている選択情報とに基づいて論理単位グループが選択され、選択された論理単位グループより文書が構成されて表示される。したがって、バージョン情報、更新日時情報、利用者情報等といった選択情報を含む選択条件を入力することによって、当該選択情報に対応する文書が構成されて表示される。

【0032】すなわち、本発明によれば、文書の全体構造と部分構造について区別なく適用できるバージョン管理を可能にし、文書の内容及び構造に自由に変更を加えても、必要なバージョンの文書を取り出すことが可能となっている。換言すれば、本発明では、文書の構造を、全体構造と部分構造のバージョンを同等に管理できるデータ構造として、文書データとしては文書の構成要素の全てのバージョンを含んだものとして、この中から適切な抽出条件によって抽出した部分構造を組み合わせて文書全体を構成する。

【0033】また、本発明によれば、共同執筆において、それぞれの担当者が与えた条件に適合するバージョンの文書を一貫して編集することが可能化し、各担当者が混乱することなく編集処理を行うことが可能になる。すなわち、本発明では、バージョン指定情報と、利用者指定情報と、日時指定情報とを組み合わせ、共同執筆者が編集の対象とするべきバージョンを取り出すための条件を定義してあり、各担当者の執筆作業の際に、対象としている文書部分にこれらの情報を関連付けておくことにより、文書を自由に編集しても、これら情報に基づいて文書を構成することにより各執筆者にとって編集すべきバージョンの文書が復元される。

【0034】

【発明の実施の形態】本発明の一実施例に係る文書処理装置及び方法を説明するに当たって、まず、本実施例における構造化文書の構成及びバージョン管理を図3～図8を参照して説明する。なお、本実施例においては、請求項に記載した、木構造を成している構造化文書の一部分である内容単位は内容部として、構造化文書の下層の構造部分との関連位置を示す参照単位はアンカーとして、実現され、更に、これら内容部とアンカーとの少なくともいずれか一方を論理単位とし、木構造の同じ階層に位置する論理単位から成る論理単位グループはストリームとして、論理単位グループを選択するための選択情報はストリーム属性として実現される。

【0035】本実施例では、ストリームに基づく木構造の構造化文書を利用しており、ストリーム単位でバージョンの管理を行うことにより文書の全体構造と部分構造を統一的に扱えるようにして、文書の全体構造と内容部とのバージョンを同等の方式によって管理することを可能にしている。すなわち、本実施例では、文書内容の変更、文書内容や文書の部分構造の挿入、文書内容や文書の部分構造の削除、文書内容や文書の部分構造の順序の変更、文書内容や文書の部分構造の移動、等といった一般的な文書構造の編集に係る基本操作と、適切な内容部やストリームの生成の操作を組み合わせることによって、任意の文書構造の生成、変更、復元等が可能になっている。

【0036】図3に示す文書構造の例では、ノードAを文書ルートのアンカーとすると、例えば、アンカーB及び内容部(u)を含むストリームaが章、アンカーC及びDを含むストリームbが節、内容部(v)と(x)を含むそれぞれのストリームc、dが項に該当することとなる。そして、新たな内容部を作成して既存の内容部と入れ替えるといった編集処理を行う場合には、例えば図4～図8に示すようなストリーム単位での処理を行って文書の内容部及び全体構造を変更する。

【0037】まず、図3に示した元の文書に対して内容部を変更する場合には、図4に示すように、新たな内容部(y)を含むストリームd2を作り、既存の内容部

(x)を含むストリームd1と新たなストリームd2とを入れ替えるとともに、古いストリームd1と新しいストリームd2とをアンカーDが参照する集合として記憶管理する。

【0038】また、図3に示した元の文書に対して新たな部分構造を挿入する場合には、図5に示すように、アンカーC及びDを含む既存のストリームb1に対して更にアンカーEを含む新たなストリームb2を作るとともに、内容部(z)を含んだ新たなストリームeを作り、新たなストリームb2を既存のストリームb1と入れ替えるとともに、新たなストリームb2の各アンカーC、D、Eによってそれぞれストリームc、d、eを参照させる。なお、この場合にあって、同一のアンカーBで参照される同じ階層のストリームb1、b2はアンカーBが参照する集合として記憶管理する。

【0039】また、図3に示した元の文書に対して既存の部分構造Dを削除する場合には、図6に示すように、アンカーC及びDを含む既存のストリームb1に対してアンカーDを削除した新たなストリームb3を作り、既存のストリームb1と新たなストリームd3とを入れ替えるとともに、古いストリームb1と新しいストリームb3とをアンカーBが参照する集合として記憶管理する。また、新たなストリームb3のアンカーCでストリームcを参照させるとともに、ストリームdを古いストリームb1のアンカーDによって参照されるストリームとして記憶管理する。

【0040】また、図3に示した元の文書に対して部分構造C、Dの順序を入れ替える場合には、図7に示すように、アンカーC及びDを含む既存のストリームb1に対してアンカーCとDの順序を入れ替えた新たなストリームb4を作り、既存のストリームb1と新たなストリームd4とを入れ替えるとともに、古いストリームb1と新しいストリームb4とをアンカーBが参照する集合として記憶管理する。

【0041】また、図3に示した元の文書に対して部分構造Dを移動させる場合には、図8に示すように、アンカーC及びDを含む既存のストリームb1に対してアンカーDを削除した新たなストリームb3を作り、既存のストリームb1と新たなストリームd3とを入れ替えるとともに、内容部(u)及びアンカーBを含む既存のストリームa1に対して更にアンカーDを含ませた新たなストリームa2を作り、既存のストリームa1と新たなストリームa2とを入れ替え、アンカーDにはそのままストリームdを参照させる。なお、この場合にあって、同一のアンカーAで参照される全てのストリームa1、a2はアンカーAが参照する集合として記憶管理し、同一のアンカーBで参照される全てのストリームb1、b3はアンカーBが参照する集合として記憶管理する。

【0042】上記のように、編集処理されたストリーム

は編集前のストリームと同一の集合として記憶され、これら集合化されたストリームは同一のアンカーによって参照される。したがって、後述するように、アンカーで示される集合の中から所定の条件に適合するストリームを選択し、これらストリームによって文書を構成することにより、編集処理によって文書の全体構成を変更した場合であっても、編集前の過去の文書を復元することができる。

【0043】このような文書処理を実行する本実施例の文書処理装置は、図1に示す構成を有している。半導体メモリや磁気記憶装置等から構成されたメモリ1には、論理単位保持手段2、論理単位グループ保持手段3、選択情報対応付け手段4が構成されており、これら手段2～4によって構造化文書に関する情報を記憶管理する。論理単位保持手段2はアンカー又は内容部の少なくともいずれか一方を含む論理単位を保持し、論理単位グループ保持手段3は論理単位から成るストリームを保持する。

【0044】選択情報対応付け手段4は、論理単位グループ保持手段3に保持されているストリームとストリームを選択するためのストリーム属性とを対応付けており、これによって、ストリーム属性を指定することにより、論理単位グループ保持手段3から対応するストリームを選択することが可能となっている。したがって、構造化文書は図2に示すように各ストリーム毎にストリーム属性を対応付けたデータ構造としてメモリ1に格納されており、本実施例における以下の説明では、これらストリームとストリーム属性との組をストリームセットとして定義する。また、ストリームセットは、同一のアンカーで参照され且つ文書構造中の同じ階層に位置するストリームとストリーム属性との組を複数個包含する場合もあり、この場合にはストリームセットは複数のストリームを包含した集合となる。また、文書構造の最上位のノードはルートアンカーであり、このルートアンカーから下位のストリームセットへ、当該ストリームセットに含まれているアンカーから更に下位のストリームセットへといったように、これらストリームセットはアンカーによる参照によって木構造化されている。

【0045】文書編集手段5は、キーボードやマウス等といった入力手段からの利用者による指示に基づいて、メモリ1に保持されている構造化文書に編集処理を施し、図4～図8に示したような内容部や部分構造（ストリーム）に対する変更処理を行う。論理単位グループ生成手段6は、文書編集手段5によって編集処理がなされたことに基づいて対応する新たなストリームを生成する処理を行い、例えば図4に示したように、内容部（x）を内容部（y）に変更した場合には、内容部（y）を含む新たなストリームd2を生成する。

【0046】参照関連付け手段7は論理単位グループ生成手段6によって新たに生成されたストリームを編集対

象となった元のストリームと同一の集合に追加して論理単位グループ保持手段3に保持させる処理を行い、例えば図4に示したように、既存のストリームd1に対して新たなストリームd2が生成された場合には、当該ストリームd2をストリームd1と同じ集合（ストリームセット）に追加する。

【0047】条件指定手段8は、利用者が操作するキーボードやマウス等の入力手段を有し、ストリーム属性を含む選択条件が入力されるとこれを論理単位グループ選択手段9へ出力する。なお、本実施例では文書のバージョン情報、文書の更新日時情報、文書の利用者情報等がストリーム属性として設定され、このようなストリーム属性を含む選択条件が論理単位グループ選択手段9に入力される。論理単位グループ選択手段9は、条件指定手段8から入力された選択条件に基づいて論理単位グループ保持手段3から選択条件に適合するストリームを選択する処理を行い、この選択処理を行うために検索手段10と選択手段11とを有している。

【0048】検索手段10は論理単位グループ保持手段3に保持されているストリーム中に選択条件に適合するストリームが存在するか否かを検索処理し、選択手段11はこの検索によって適合するストリームが存在するときには当該ストリームを選択する処理を行う。例えば、選択条件として文書のバージョンと日時情報と利用者情報が指定された場合、検索手段10によって、バージョン情報に対応する更新日時情報以降で、指定された日時情報以前の更新日時情報と対応付けられ、且つ、指定された利用者情報に一致するストリームを検索し、更に、選択手段11によって、この選択条件に適合するストリームを選択する。なお、選択手段11は検索手段10による検索によって、上記のように選択条件に含まれる全ての属性に適合するストリームが存在しない場合には、この属性の内からバージョン情報に一致するストリームを選択する。

【0049】文書構成手段12は論理単位グループ選択手段9によって選択されたストリームに基づいて文書を構成する処理を行い、文書構成表示手段13は文書構成手段12によって構成された文書をディスプレイ14に表示させる処理を行う。未表示情報表示手段15は、文書構成表示手段13によって文書の構成部分として表示されていない未表示のストリームをマーカとしてディスプレイ14に表示させる処理を行う。すなわち、論理単位グループ選択手段9によって選択されたストリームを含むストリームセットが他のストリームを含んでいる場合には、当該他のストリームを後述（図17）するようにマーカ化した候補として、文書構成手段12によって構成された文書とともにディスプレイ14に表示する。

【0050】論理単位グループ指定手段16は、キーボードやマウス等の入力手段を有し、利用者からの入力に基づいて未表情報表示手段15によって表示されたマー

10

20

30

40

50

カを指定する処理を行う。論理単位グループ再選択手段17は、論理単位グループ指定手段16によってマーカを指定したことに基づいて、論理単位グループ選択手段9に、当該指定されたマーカに対応するストリームと論理単位グループ選択手段9で既に選択されているストリームとを入れ替える処理を行わせる。すなわち、ディスプレイ14に或るストリームの内容部（テキスト）が文書の構成部分として表示されている場合、マーカを選択することによって、他のストリームの内容部（テキスト）が入れ替えて表示される。また、入れ替えられたストリーム間でアンカーの数や位置等といったアンカーに係る構成が異なる場合には、表示されている文書の全体構成も変更される。

【0051】ここで、上記した論理単位グループ選択手段9が行う選択条件に基づくストリームの選択処理を図9を参照して更に詳しく説明する。なお、図9に示す例では、各ストリームの属性として、“Usui”、“Sato”、“Tani”といった利用者名からなる利用者情報と、更新日時情報と、バージョン情報とが対応付けられており、入力される選択条件には、文書を参照して処理を行うための基準日時（基準日時情報）と、基準となるバージョン情報（基準バージョン情報）と、利用者情報と、が含まれている。また、縦軸の日時に沿って利用者情報毎のストリーム（1）～（5）を示してある。

【0052】この例では、利用者の“Usui”さんが参照日時として図中に点線で示した時点で実際に文書を参照する場合を考える。まず、選択条件として何も指定しない場合には、参照日時を基準として最新のストリーム（5）が選択される。また、“本人”（すなわち、“Usui”）とした利用者情報のみを指定した場合には、ストリーム属性が“Usui”である最新のストリーム（1）が選択される。また、“Sato”とした利用者情報のみを指定した場合には、ストリーム属性が“Sato”である最新のストリーム（5）が選択される。また、図中に示す基準日時のみを指定した場合には、ストリーム属性の更新日時が基準日時までの最新のストリーム（4）が選択される。

【0053】また、基準日時として図示された日時を指定し、且つ、利用者情報として“Sato”を指定した場合には、ストリーム属性が“Sato”で且つ更新日時が基準日時までの最新のストリーム（3）が選択される。また、図示された基準バージョンのみを指定した場合には、基準バージョン以降の最新のバージョン（5）が選択される。また、図示された基準バージョンと基準日時として図示された日時を指定した場合には、基準バージョン以降で、基準日時までの最新のストリーム（4）が選択される。

【0054】また、図示された基準バージョンと基準日時として“最初”を指定した場合には、基準バージョン

以降で、最初のストリーム（2）が選択される。また、図示された基準バージョンと、利用者情報として“本人”を指定した場合には、基準バージョン以降にはストリーム属性が“Usui”であるストリームがないため、基準バージョンのストリーム（2）が選択される。このように、ストリーム属性に基づいて、条件指定手段8から指定された選択条件に適合するストリームが選択され、当該ストリームに基づいて文書構成手段12によって文書が構成される。

10 【0055】次に、上記構成の文書処理装置によって実施される文書編集処理の概要を図10～図36に示す具体例を用いて説明する。なお、以下の説明によって、上記した各構成手段による機能も更に明確化されることとなる。この具体例では、図10に示す編集課程によってSatoさんとTaniさんが「発明説明書」という名前の文書を共同で執筆している。なお、個々の編集作業の開始から終了までをセッションと称し、開始は文書編集ツールで文書を開くことによって行われ、終了は文書を閉じることによって行われる。

20 【0056】図10では、横軸に時間を取り、1～7の各セッションは担当者“Sato”、“Tani”に対応する水平方向の線分で示される。また、各線分の後端の黒い三角形は文書の保存が行われた時点を示しており、また、保存のマークからの垂直方向の線でセッションの名前を定義したことを示してある。例えば、セッション4では、1995年1月4日に文書を保存し、この際に「中間報告」という名前のバージョンを定義していることを示している。なお、この具体例では、それぞれのセッションは、時間が重ならないようにしているが、これは説明を無用に複雑にしないためであり、本発明はセッションの時間が重なっても何らの支障を生じるものではない。

30 【0057】利用者Satoさんが新しく文書「発明説明書」を作成して保存するセッション1について、図11～図15を参照して説明する。まず、図11に示すように、1995年1月1日の15:00に、Satoさんが文書編集手段5の文書エディタを起動すると、ディスプレイ14に図に示す対話ツールが表示されて新規文書名の入力を利用者に促す。これに対して、Satoさんが文書ルート名として「発明説明書」を入力し、設定完了を指示する「ok」ボタンをクリックすると、文書の抽出を開始するルートアンカーの名前が設定される。なお、このルートアンカーは文書を構成するストリームセットを参照するものである。

40 【0058】次いで、図12に示すように、文書エディタが抽出した文書構造をディスプレイ14に表示する。この文書構造の表示は、アンカーの位置を示す白い三角形のマーカと、アンカー名と、アンカーの参照するストリームの内容部とを表示することにより行われる。なお、アンカーの位置を示すマーカのインデントの深さ

性、が選択情報対応付け手段4によって対応付けられる。

【0067】次に、図10に示したセッション3においては、Satoさんが自分がセッション2で編集した文書「発明説明書」に更に編集を加え、編集結果をメモリ1に保存する。まず、1995年1月3日の15:00に、セッション2と同様にしてSatoさんが文書「発明説明書」の編集を開始すると、文書エディタはセッション2で保存したときの文書構造をメモリ1から抽出して図20(1)に示すようにディスプレイ14に表示する。

【0068】これに対して、Satoさんが、文書の編集作業として、図20(1)に示すように請求項の欄に項目2を追加すると、論理単位グループ生成手段6が請求項のアンカーBが参照するストリームとして、項目1のアンカーCと項目2のアンカーDを含むストリームを作成する。そして、図20(2)に示すように、この項目2にテキストを記述すると、このテキストからなる内容部(x)がアンカーDによって参照される新たなストリームとして論理単位グループ生成手段6によって生成される。

【0069】したがって、文書エディタ内には図21に示すように、ルートアンカーAによって参照されて内容部(u)とアンカーBとを含むストリームa2と、アンカーBによって参照されてアンカーCとDとを含むストリームb2と、アンカーCによって参照されて内容部(v)を含むストリームc1と、アンカーDによって参照されて内容部(x)を含むストリームd1と、が生成される。なお、上記の編集処理によってアンカーBが参照するストリームが複数の候補となるので、図20に示すように、これらストリームb1、b2の内の未表示のストリームを示すマーカMbがディスプレイ表示される。

【0070】次いで、編集処理を終了して、1995年1月3日の20:00に、Satoさんがセッション2と同様にして文書を保存する指示を入力すると、メモリ1には図22に示す文書構造が格納される。すなわち、この文書構造には、図19に示した構造に対して、ストリームセットbにアンカーCとDを含むストリームb2が追加され、更に、アンカーCで参照されるストリームセットc、ストリームセットcに含まれる内容部(v)を有したストリームc1、アンカーDで参照されるストリームセットd、ストリームセットdに含まれる内容部(x)を有したストリームd1、が追加される。更に、この文書構造の新たに追加されたストリームb2、c1、d1に対しては、利用者情報の”Sato”、データ保存(更新)日時情報の「1995. 1. 3 20:00」といったストリーム属性が選択情報対応付け手段4によってそれぞれ対応付けられる。

【0071】次に、図10に示したセッション4におい

ては、Satoさんが作成した文書「発明説明書」にTaniさんが編集を加え、その編集結果を「中間報告」バージョンとして保存する。まず、1995年1月4日の10:00にTaniさんがセッション2と同様にして編集を開始すると、図23に示すように抽出条件設定ツールがディスプレイ14に表示される。これに対して、Taniさんは設定条件として何も指定せずに「ok」ボタンをクリックして、図24に示すような文書エディタをディスプレイ表示させて編集を実行する。なお、このように何も条件を設定せずに編集を開始すると、利用者に関わらず最新の文書がメモリ1から抽出されて編集対象となる。また、ここで、Taniさんはこれまでに、文書の更新を行っていないので、Satoさんのように利用者条件として条件指定手段8から「%本人」を指示すると、文書として何も抽出されないことになる。

【0072】図24に示すように、文書エディタはセッション3で保存したときの文書構造を抽出してディスプレイ表示しており、Taniさんが編集作業として、図に示すように請求項のアンカーBが参照するストリームとして、項目1と項目2の間に項目1.5を追加する。この編集の結果、図25に示すように、論理単位グループ生成手段6によって、アンカーBによって参照されてアンカーC、D、Eを含む新たなストリームb3、アンカーEによって参照される項目1.5の内容部(z)を含む新たなストリームe1が生成される。

【0073】次いで、編集処理を終了して、1995年1月4日の15:00に、Taniさんが文書エディタの「save」ボタンをクリックし、文書を保存する指示を入力すると、ディスプレイ14には図26に示す対話ツールが表示されて保存時の指定を促す。これに対して、Taniさんはバージョン情報として「中間報告」を設定して、「ok」ボタンをクリックして作成した文書の保存処理を指示する。

【0074】この保存処理の結果、メモリ1には、図27に示す文書構造が格納される。すなわち、図22に示したセッション3での文書構造に対して、アンカーBで参照されるストリームセットbにアンカーC、D、Eを含む新たなストリームb3が追加され、更に、アンカーEによって参照されるストリームセットe、及び、ストリームセットeに含まれる新たなストリームe1、が追加される。なお、この編集処理によって生成されたストリームb3、e1には、ストリーム属性として、編集した利用者の情報である”Tani”、データを保存(更新)した日時である「1995. 1. 4 15:00」、バージョン情報である「中間報告」が選択情報対応付け手段4によって対応付けられる。また、この編集処理に関わったストリームa2、c1、d1のストリーム属性にも、選択情報対応付け手段4によってバージョン情報の「中間報告」が追加して対応付けられる。

【0075】次に、図10に示したセッション5においては、Satoさんが自分が編集した文書「発明説明書」に更に編集を加え、編集結果を保存する。なお、この編集開始時にはセッション4でTaniさんによる変更が加えられているが、抽出条件の設定により、自分が編集して保存した状態の文書について編集を再開する。すなわち、セッション2と同様にして1995年1月4日の15:00に文書エディタ起動する。そしてこの際に、条件指定手段8から入力する抽出条件設定ツールの指定を「%本人」とすることによって、論理単位グループ選択手段9は請求項のアンカーBが参照するストリームとして、Taniさんが作成したb3ではなく、Satoさんの作成したb2を選択し、セッション3でSatoさんが保存した状態の文書構造を図28(1)に示すように文書エディタで表示する。

【0076】なお、この場合の候補マーカMbは反転表示され、これによって、請求項のアンカーBが現在参照しているストリームの候補により新しいストリームがあることを示す。そして、Satoさんが編集作業として、図28(2)に示すように、項目2の内容部(x)を内容部(y)に書き換えると、図29に示すように、論理単位グループ生成手段6が項目2のアンカーDが参照するストリームとして、内容部(x)を内容部(y)に書き換えたストリームd2を新たに生成する。

【0077】次いで、1995年1月4日の20:00にSatoさんがセッション1と同様にして文書を保存すると、メモリ1には図30に示す文書構造が格納される。すなわち、図27に示したセッション4での文書構造に対して、アンカーDで参照されるストリームセットdに内容部(y)を含む新たなストリームd2が追加され、更に、このストリームd2にはストリーム属性として、編集した利用者の情報である”Sato”、データを保存(更新)した日時である「1995. 1. 4 20:00」、が選択情報対応付け手段4によって対応付けられる。

【0078】次に、図10に示したセッション6においては、Taniさんが自分が編集を加えて保存した文書「発明説明書」の「中間報告」バージョンと、その後、自分以外の執筆者によって加えられた編集の結果を参照する。1995年1月5日の10:00に、Taniさんが文書「発明説明書」の編集を開始し、セッション2と同様にして、文書フォルダのアイコンを選択してダブルクリックすることで、文書エディタを起動する。この時、図31(1)に示す抽出条件設定ツールによって、文書の抽出条件の設定が促され、これに対して、Taniさんがバージョン情報として「中間報告」、利用者情報として「%本人」を指定する。この指定によって「中間報告」として指定したバージョンに、Taniさんにより更新された構成要素(ストリーム)から成る文書がメモリ1から抽出されて編集対象となる。

【0079】すなわち、論理単位グループ選択手段9が条件指定手段8から入力された条件に従って、「中間報告」を属性に含むストリームを検索し、Taniさんの作成した最新のストリームを選択する。ここで、Taniさんがセッション4でバージョンを設定後、Satoさんは編集を行っているが、Taniさんは編集を行っていないため、項目2のアンカーDが参照するストリームとして、Satoさんの作成した最新のストリームd2ではなく、「中間報告」を設定した時点でのストリームd1が選択される。この結果、図32(1)に示すように、文書エディタにはTaniさんがセッション4で保存した「中間報告」の文書構造(図24)が表示される。なお、項目2のアンカーDに対応する候補マーカMdは反転表示されて、Satoさんによって作成されたより新しいストリームの候補があることを示している。

【0080】ここで、文書エディタの「filter」ボタンをクリックすることによって、抽出条件設定ツールを再び呼び出すことができ、この呼び出したツールで条件を変更すると、新しい条件で文書構造が再抽出される。したがって、Taniさんが図31(2)に示すように、抽出条件として「%本人」を除いた指定を行うと、この指定によって、より新しいストリームが利用者の区別なく選択される。この結果、項目2のアンカーDが参照するストリームとして、Satoさんが作成した最新のストリームd2が選択され、図32(2)に示す文書が文書エディタに表示される。

【0081】したがって、図31(1)に示した最初の条件の指定では、エディタ内には図33(1)に示す文書構造が取り出され、図31(2)に示した条件の変更後は、図33(2)に示す文書構造が取り出される。すなわち、図33(1)に示す文書構造ではアンカーDが内容部(x)を含むストリームd1を参照するが、図33(2)に示す文書構造ではアンカーDが内容部(y)を含むストリームd2を参照する。なお、セッション6ではTaniさんは編集を終了するが、文書は保存せず、メモリ1に格納される文書構造は図30に示した構造のままである。

【0082】次に、図10に示したセッション7においては、Satoさんが自分が編集を行ってきた文書「発明説明書」について、自分以外の執筆者による編集の結果を反映させたものを「最終報告」バージョンとして保存する。セッション7では、1995年1月5日の15:00に、Satoさんがセッション2と同様にして文書エディタを起動させ、文書「発明説明書」の編集を開始する。

【0083】文書エディタは、図34に示すように、セッション5でSatoさんが保存したときの文書構造をメモリ1から抽出してディスプレイ表示する。この文書エディタには請求項のアンカーBが参照するストリームに、未選択のより新しい候補があることが候補マーカM

bによって示されている。この候補マーカMbをクリックすると、候補のストリームb1、b2、b3がメニューKとして表示され、このメニューKから論理単位グループ指定手段16でストリームを選択すると、論理単位グループ再選択手段17によって選択されたストリームが現在参照中のストリームと置き換えられ、更に、置き換えられたストリームより下位の構造は抽出条件に従って再構成される。

【0084】したがって、Sat oさんが候補メニューKから最新のストリームb3を選択することによって、セッション5で作成したと同じ文書を編集対象とできる。このセッション7の編集処理では、文書エディタ内で、図29に示したセッション5での文書構造が図33(2)に示した構造に変更される。そして、1995年1月5日の20:00に、Sat oさんが文書を保存する指示を行うと、図35に示す対話ツールがディスプレイ表示されて、保存時の条件設定を促す。これに対して、Sat oさんがバージョン情報として「最終報告」を指定して保存を行うと、メモリ1には図36に示すように編集処理が加えられた文書構造が格納される。すなわち、図30に示したセッション5での文書構造に対して、保存処理時に参照していたストリームa2、b3、c1、d2、e1の属性に、選択情報対応付け手段4によってバージョン情報の「最終報告」が追加して対応付けられる。

【0085】次に、文書をメモリ1から抽出してディスプレイ14に表示する処理を図37～図40に示すフローチャートを参照して更に詳しく説明する。図37に示すように、まず、利用者による条件指定手段8からの指示に従って論理単位グループ選択手段9がメモリ1からルートアンカーを選択し(ステップS1)、引き続いて、指定されたルートアンカーの下位の階層に位置する構造をメモリ1から抽出して文書をディスプレイ14に表示する処理を行う(ステップS2)。

【0086】このステップS2での下位構造抽出表示処理は図38に示す手順でなされる。まず、文書構成手段12が抽出中の文書構造にルートアンカーを組み入れ(ステップS11)、文書構造表示手段13がこのルートアンカー名を表示した文書エディタをディスプレイ14の画面に表示する(ステップS12)。次いで、ルートアンカーが参照しているストリームセットを論理単位グループ選択手段9がメモリ1から選択し(ステップS13)、このストリームセットから、利用者によって条件指定手段8から指示された条件に適合する属性のストリームを選択する(ステップS14)。

【0087】次いで、選択されたストリームを文書構成手段12が抽出中の文書構造に組み入れ(ステップS15)、文書構造表示手段13がこのストリーム(内容部)をディスプレイ14の画面に表示されている文書エディタ中表示する(ステップS16)。次いで、未表

示情報表示手段15が選択されたストリームが上記のストリームセット内の唯一のストリームかを判断し(ステップS17)、ストリームセット内に他のストリームが無いときには処理を終了する。なお、ストリームに含まれるアンカーが下位の階層のストリームセットを参照している場合には、これらストリームセット及びストリームについて上記と同様な選択表示処理が繰り返し行われる。

【0088】一方、選択されたストリームの他にストリームセット内に他のストリームが存在する場合には、未表示情報表示手段15が当該他のストリームは選択したストリームより新しいストリームであるかをそのストリーム属性から判断し(ステップS18)、より新しいストリームである場合には当該ストリームを表す候補マークをディスプレイ14上の文書エディタ中に反転表示する一方(ステップS19)、新しいストリームでない場合には当該ストリームを表す候補マークをディスプレイ14上の文書エディタ中に非反転で表示する(ステップS20)。

【0089】ここで、上記のストリーム表示処理(ステップS16)を更に詳しく説明すると、当該処理は図39に示す手順で実行される。まず、選択されたストリームが空のストリームであるかを判断し(ステップS31)、内容部やアンカーといった論理単位を含まない空のストリームである場合にはそのまま処理を終了する。一方、ストリームに論理単位が含まれている場合には、これら論理単位の内未処理のものを1つ選択し(ステップS32)、当該選択した論理単位がアンカーであるかを判断する(ステップS33)。

【0090】この判断の結果、アンカーである場合には図38に示した下位構造抽出表示処理を行ってアンカーで参照されている下位構造の表示を行う一方(ステップS34)、内容部である場合には当該内容部を表示処理する(ステップS35)。そして、選択されたストリーム中の全ての論理単位について上記の処理が終了したかを判断し(ステップS36)、終了していない場合にはステップS32以降の処理を繰り返し行い、終了している場合には当該ストリーム表示処理を終了する。

【0091】また、上記のストリーム選択処理(ステップS14)を更に詳しく説明すると、当該処理は図40に示す手順で実行される。まず、ストリームセット中の全てのストリームを候補リストにし(ステップS41)、条件指定手段8から入力された条件にバージョン情報の指定があるかを判断する(ステップS42)。この判断の結果、バージョン指定がある場合には、候補リスト中のストリームから条件に適合する属性を有したストリームを探し、このストリームを一旦基準ストリームとして設定する(ステップS43)。

【0092】次いで、条件指定手段8から入力された条件に基準日時情報の指定があるかを判断し(ステップS

10

20

30

40

50

は、文書構造中におけるストリームの階層の深さを示しており、下層のストリームを参照するアンカーになる程、マーカの表示される位置は図中の右方向となる。

【0059】すなわち、「発明説明書」という名称のルートアンカーで参照されるストリームに未だ内容部（テキスト内容）が未だ含まれていないときには、図12

(1)に示すように、ルートアンカー名称「発明説明書」が表示され、当該ルートアンカーが参照する空のストリームが図中に点線で示す領域として表示される。そして、Satoさんが編集作業を行って、ルートアンカーが参照するストリームに内容部(u)としてテキストを記述すると、このテキスト内容が図12(2)に示すように上記の領域内に表示される。この編集の結果、論理単位グループ生成手段6によって、図13に示すように、新しく作成された内容部(u)だけからなるストリームaが生成され、このストリームaがルートアンカーAによって参照された文書構造が構成される。

【0060】次いで、編集処理を終了して、1995年1月1日の20:00に、Satoさんが図12に示す文書エディタの「save」ボタンをクリックし、文書を保存する指示を入力すると、ディスプレイ14には図14に示す対話ツールが表示されて保存時の指定を促す。これに対して、この例では、Satoさんは何も設定することなく、「ok」ボタンをクリックして作成した文書の保存処理を指示する。なお、この対話ツールには文書ルートとしてルートアンカー名「発明説明書」は表示されている。

【0061】この保存処理の結果、メモリ1には、図15に示す文書構造が格納される。すなわち、この文書構造には、ルートアンカーA、ルートアンカーAで参照されるストリームセットa、ストリームセットに含まれるストリームa1、ストリームa1に含まれる内容部(u)が新しく生成され、ストリームa1にはストリーム属性として、編集した利用者の情報である“Sato”、データを保存(更新)した日時である「1995.1.1 20:00」が選択情報対応付け手段4によって対応付けられる。

【0062】次に、図10に示したセッション2においては、Satoさんが自分がセッション1で作成した文書「発明説明書」に編集を加え、編集結果をメモリ1に保存する。まず、図16(1)に示すようにディスプレイ14には文書フォルダが表示されており、1995年1月2日の15:00に、Satoさんが文書フォルダの「発明説明書」アイコンをダブルクリックして選択すると、文書編集手段5の文書エディタが起動される。この結果、ディスプレイ14には図16(2)に示す抽出条件設定ツールが表示され、抽出条件の設定を促す。これに対して、Satoさんが条件指定手段8から利用者情報として本人を表す変数「%本人」を入力し、設定完了を指示する「ok」ボタンをクリックすると、編集処

理が開始される。なお、「%本人」「%最初」等の

「%」記号は、文書抽出を開始する時に、システムが値を計算して設定する変数であることを示している。

【0063】上記の設定は、本人が更新した最新の構成部分(ストリーム)を含む文書を抽出して編集対象とすることを示しており、図17(1)に示すように、文書エディタが文書エディタは、セッション1で保存したときの文書構造をメモリ1から抽出してディスプレイ14に表示する。次いで、Satoさんが文書の編集作業として、図17(1)に示すようにルートアンカーである「発明説明書」が参照するストリームに、アンカーとして「請求項」を追加し、更に、アンカー「請求項」が参照するストリームに内容部(v)としてテキストを記述すると、このテキスト内容が図17(2)に示すようにアンカー「請求項」が参照する領域内に表示される。

【0064】この編集の結果、図18に示すように、論理単位グループ生成手段6によって、ルートアンカーAによって参照されて内容部(u)とアンカーBを含む新たなストリームa2が生成されるとともに、アンカーBによって参照されて内容部(v)を含む新たなストリームb1が生成される。なお、図17に示すように、文書エディタ中には未表示情報表示手段15によって候補マーカMaが表示され、アンカーAが参照するストリームセットにはディスプレイ表示されているストリームの他に、他のストリームが候補として存在することを示している。この場合、アンカーAが参照するストリームの候補にはa1とa2とがあり、一方のストリームについてディスプレイ表示されているときには、他方のストリームはマーカMaによってディスプレイ表示される。

【0065】次いで、編集処理を終了して、1995年1月2日の20:00に、Satoさんが図17に示す文書エディタの「save」ボタンをクリックし、文書を保存する指示を入力すると、ディスプレイ14には図14に示したと同様な対話ツールが表示されて保存時の指定を促す。これに対して、Satoさんがセッション1のときと同様にして文書の保存処理を指示すると、メモリ1には、図19に示す文書構造が格納される。

【0066】すなわち、この文書構造には、ルートアンカーA、ルートアンカーAで参照されるストリームセットa、ストリームセットaに含まれるストリームa1及びa2、アンカーBによって参照されるストリームセットb、ストリームセットbに含まれるストリームb1、が含まれ、更に、ストリームa1には利用者情報の“Sato”、データ保存(更新)日時情報の「1995.1.1 20:00」といったストリーム属性、ストリームa2には利用者情報の“Sato”、データ保存(更新)日時情報の「1995.1.2 20:00」といったストリーム属性、ストリームb1には利用者情報の“Sato”、データ保存(更新)日時情報の「1995.1.2 20:00」といったストリーム属

44)、日時指定がある場合には、指定された日時より新しい日時が属性に設定されているストリームを候補リストから削除する(ステップS45)。すなわち、利用者が処理対象としている日時以降に編集処理されたストリームを除外する。次いで、条件指定手段8から入力された条件に利用者情報の指定があるかを判断し(ステップS46)、利用者の指定がある場合には、指定された利用者情報が属性に設定されていないストリームを候補リストから削除する(ステップS47)。すなわち、利用者が処理対象としている自己の編集に係るストリーム以外を除外する。

【0093】次いで、基準ストリームが選択されているかを判断し(ステップS48)、選択されている場合には、基準ストリームよりも古いストリームを候補リストから削除し(ステップS49)、候補リストが空となっていないことを確認して(ステップS50)、基準ストリームを条件指定手段8から入力された条件に適合するストリームとして選択する(ステップS54)。一方、上記の判断の結果、候補リスト中に基準ストリームが残存していない場合には、候補リストが空となってしまうかを判断し(ステップS52)、空となっている場合には、いずれのストリームの選択されないことを示すために、空のストリームを選択されたストリームとする(ステップS53)。

【0094】また、上記の判断において、基準ストリームが選択されておらず、候補リストは基準ストリームが空ではない場合(ステップS52)又は候補リストから基準ストリームより古いストリームを削除しても空ではない場合(ステップS50)には、候補リスト中の最新のストリームを選択されたストリームとする(ステップS54)。したがって、ストリーム属性に基づいて、条件指定手段8から入力された条件に適合するストリームが選択される。

【0095】次に、利用者が文書エディタ中に表示された候補マーカMa、Mb、・・・を選択することによって、ストリームを置き換えた文書をメモリ1から抽出してディスプレイ14に表示する処理を、図41に示すフローチャートを参照して更に詳しく説明する。まず、論理単位グループ指示手段16によって利用者が候補マーカを指示すると、未表示情報表示手段15が指示されたストリームが属するストリームセットをメモリ1に格納されている文書構造から選択し(ステップS61)、選択したストリームセット中の各ストリームを候補メニューKとして文書エディタ中に表示する(ステップS62)。

【0096】そして、利用者が論理単位グループ指示手段16によって指示して、候補メニューK中の現在選択表示されているストリームとは別のストリームを選択したかを判断し(ステップS63)、別のストリームを選択した場合には、現在文書エディタで表示しているストリ

ームの下位階層の構造を論理単位グループ再選択手段17が破棄し(ステップS64)、更に、再選択されたストリームを破棄したストリームと入れ替える(ステップS65)。そして、この入れ替えられたストリームから成る文書を文書構成手段12が構成し、当該文書を表示手段13が文書エディタ中の表示し直して処理を終了する(ステップS66)。

【0097】次に、編集処理した文書をメモリ1に保存する処理を図42に示すフローチャートを参照して更に詳しく説明する。まず、文書編集手段5によって文書エディタで編集した文書構造のルートアンカーがメモリ1に格納されている文書構造中のルートアンカーリストに含まれているかを判断する(ステップS71)。この結果、ルートアンカーリストにない場合には、編集された文書は全く新規な文書であるので、当該文書構造のルートアンカーを参照関連付手段7がルートアンカーリストに追加する(ステップS72)。

【0098】次いで、文書エディタで生成された内容部を参照関連付手段7が所定のストリームに関連付け(ステップS73)、生成されたストリームを集めて(ステップS74)、以下の処理が未だなされていないストリームを選択する(ステップS75)。次いで、未処理のストリームが対応するストリームセットをメモリ1に格納されている文書構造中から探し(ステップS76)、対応するストリームセットがある場合には(ステップS77)、当該ストリームセットに未処理のストリームを追加する(ステップS79)。一方、対応するストリームセットがない場合には、新たなストリームセットを文書構造中の追加して(ステップS78)、当該ストリームセットに未処理のストリームを追加する(ステップS79)。

【0099】そして、ストリームセットに追加したストリームに対して、日時や利用者といった属性を選択情報対応付け手段4が設定する(ステップS80)。上記の一連の処理は、生成された全てのストリームについて繰り返し行われ(ステップS81)、この後に、ストリーム属性としてのバージョン情報の設定処理が行われる。すなわち、編集保存処理においてバージョン情報の設定がなされているかを判断し(ステップS82)、設定がなされている場合には、メモリ1に格納されている文書構造中の全てのストリームを集めて(ステップS83)、以下の処理を行っていない未処理のストリームを順次選択する(ステップS84)。

【0100】そして、選択したストリームが属するストリームセットを文書構造中から探し(ステップS85)、当該ストリームセット中の選択したストリームの属性にバージョン情報を追加する(ステップS86)。そして、上記ステップS84以降の処理を全てのストリームについて終了したところで、文書保存処理を終了する(ステップS87)。したがって、編集処理で生成さ

れたストリームを保存するに際して、当該ストリームは編集前のストリームと同一のストリームセットに追加され、また、保存処理に際して設定された属性がストリームに対応付けられる。

【0101】なお、上記の実施例で採用したアルゴリズムは、ルートアンカーから順にアンカーに出会ったときに、参照しているストリームセットから選択条件に適合するストリームの選択を逐次行って文書を生成するものであるが、これ以外にも、文書構造に含まれるストリームセットの全てから選択条件に適合する全てのストリームを選択した後で、これら選択したストリームを組み立てて文書を生成するようにしてもよい。また、上記の実施例では、文書エディタの起動時に抽出条件を必ず指定しておくようにしているが、特に指示のないときには、前回の指定と同じにする、或いは、デフォルトの条件にするようにしてもよい。この場合、複数の担当者が自分の編集を一貫して行えることを最も重視するならば、デフォルトの条件は利用者指定を「%本人」にしておけばよい。また、最新の文書の編集を行えることを最も重視するならば、デフォルトの条件は何も指定しないように

【0102】また、上記の実施例では編集処理に際して文書構造中に新しいバージョンを追加するようにしたが、参照処理を行っただけでも、新しいバージョンを追加するようにしてもよい。このようにすれば、参照処理を行った時点での文書構造を取り出すことが可能になる。

【0103】また、上記の本実施例におけるストリームの選択手順ではバージョンを指定した日時よりも基準日時が新しいことを前提としているが、基準日時とバージョン日時が逆転した場合には以下のような方法を採用することにより支障なくストリームを選択処理することができる。

- (1) 基準日時よりもバージョンの指定を優先する。
- (2) バージョンを指定した日時よりも前の基準日時を指定できないようにする。なお、ストリーム属性ではバージョンを指定した日時は記録されないため、このためには、バージョンを指定した日時を記録する手段が必要である。

【0104】また、上記の実施例ではバージョンを指定した場合にストリーム属性にバージョン情報を追加するようにしたが、この代わりに、新しいストリームとして追加するようにしてもよい。また、バージョンの指定は、利用者が明示的に行っているが、例えば、プリント時等に自動的にバージョンを記録してもよい。また、上記の実施例では、「請求項」等のアンカー名はアンカーを含むストリームに対応付けて管理されている。したがって、アンカーの名前が変更されることによっても、新しいストリームが生成されるが、この他にもストリームとは独立にアンカー名を管理したり、参照しているスト

リームの名前として管理するようにしてもかまわない。

【0105】また、上記の実施例では論理単位グループとしてストリームという単位で文書の構造を表現するモデルを採用したが、本発明はこの文書構造の表現方法に限定されるものではなく、従来技術で示したようなノードとリンクからなる木構造による文書構造に本発明を適用することもできる。なお、ノードとリンクによる文書構造においては、本発明の論理単位グループに対応するのは1つのノードを起点とするリンク集合である。要は、本発明は、このリンク集合やストリームといった単位をバージョン管理の対象として、編集処理毎に複製、保存、属性設定等を適切に行う手段を与えていることに特徴がある。

【0106】例えば、従来技術である特開平6-35914号公報に記載された発明を拡張して本発明の機能を提供するためには、次のようにすればよい。末端ではないノードについて下位ノードへのリンク列をノードの内容として扱うことにし、そのノードに対応するバージョンノードのそれぞれが、異なるリンク列を持つようにする。このように拡張すると、下位ノードを本発明における参照単位、下位ノードへのリンク列を論理単位グループとして扱えるようになる。

【0107】更に言えば、本発明では、従来例として説明した多重文書処理システム（特開平7-44563号公報、特開平6-131343号公報）に示されているデータ構造を文書構造として採用することも可能であり、当該データ構造において本発明の論理単位グループに対応するものは、ノードが保持する子ノードポインタの集合となる。なお、当該多重文書処理システムではノードを多重化する手段を提供しているが、これに加えて、本発明は編集時に子ノードポインタ集合を多重化する手段を有している。すなわち、本発明では、編集を行った際にノードを多重化するだけでなく、その子ノードポインタの集合を複製、保存、属性設定を行う手段を有している。具体的には、文書の更新が生じたときに、自動的に子ノードポインタ集合を複製して管理するとともに、その属性値として日時、利用者、バージョン名を指定できるようにしている。このような処理を利用者が手作業で行うのであれば、編集作業が極めて複雑なものとなることはいうまでもなく、本発明の本質は、この複雑な処理を計算機によって自動的に且つ適切に実行することによって、複数の担当者による執筆作業における負担を大幅に軽減することにある。

【0108】また、上記の実施例では、文書構造を内容部のテキスト表示と一緒に表示する例を示したが、図43、44に示すように、テキスト内容を表示せずに構造だけを表示するようにしてもよい。すなわち、図43では、アンカーを矩形、ストリームを矢印として表示しており、他のストリームの候補があるかどうかは、ストリームを示す矢印を候補の数だけ表示することで示され

る。また、図 44 では、ストリームを参照するアンカーの木構造として表示しており、ストリームに他の候補があるかどうかは、アンカーを示す図形の右肩に候補の数だけ影をつけることで示される。

【0109】次に、本発明に係る構造化データ処理装置の一実施例を説明する。なお、上記した文書処理装置と比較すると、内容単位はノード、参照単位はリンク、論理単位グループは同一の編集処理に係る（すなわち、兄弟の関係にある）リンクの集合に該当する。また、本実施例についての以下の説明では、過去に存在していなかった構造を生成する例も示す。

【0110】ここで、まず、本実施例が処理の対象としている構造化データのバリエーションについて説明しておく。図 45～図 47 には、ノード、リンク、リンク間関係、文書内容、属性マッピング、選択属性についてのデータ保持方法のバリエーションを示してあり、ノードとリンクからなる従来技術のデータ構造と本質的に異なる点は、同じノードを始点とする複数のリンク間関係を保持し、それぞれに属性情報をマッピングすることによって、リンク間関係を管理することにある。いずれのバリエーションにおいても、これらの構造を表現するために必要な情報が管理されていることが本発明の構成上重要な要件である。

【0111】図 45 は、リンク間関係の実現方法のバリエーションを示している。同図（a）に示す構造はノードの記述中にそれを始点とする複数のリンク間関係を保持する構成であり、この構造では、リンク間関係は共通の始点ノードを規定する必要がないので、終点ノードへの参照ポインタのリストとして実現できる。なお、本実施例では後述するように、編集対象としてカレント構造管理手段 21 に保持されるカレント構造を表現するためにこの構成を採用している。また、同図（b）に示す構造はリンク間関係が始点ノードと終点ノードへの参照ポインタのリストとして表現されており、同図（c）に示す構造は共通の始点ノードを持つリンク間関係の集合をリンク間関係セットとして保持し、そこから始点ノードを参照するようになっている。なお、本実施例では後述するように、ハイパー構造管理手段 20 に保持されるハイパー構造を表現するために、同図（c）に示す構成を採用している。

【0112】図 46 は、属性マッピングの実現方法のバリエーションを示している。同図（a）に示す構造では、リンク間関係の記述が後述する選択属性を含んでいる。また、同図（b）に示す構造ではリンク間関係セットが各リンク間関係に選択属性をマッピングしており、同図（c）に示す構造では属性値の管理手段が選択属性をリンク間関係にマッピングしている。なお、本実施例では、同図（a）に示す構造を採用している。図 47 は、ノードによる文書内容の保持方法のバリエーションを示している。同図（a）に示す構造はノード中に内容

を記述する構成であり、同図（b）に示す構造はリンク間関係の記述が内容を記述する構成であり、同図（c）に示す構造は文書内容の管理手段で管理されている内容を参照する構成である。なお、本実施例では同図（b）の構成を採用している。

【0113】次いで、図 48 には本実施例に係る構造化データ処理装置の構成を示してあり、本装置によって上記構成の構造化データを処理する。構造化データ処理装置には、ハイパー構造管理手段 20、カレント構造管理手段 21、編集手段 22、登録指示手段 23、登録手段 24、履歴管理手段 25、履歴表示手段 26、抽出手段 27、抽出指示手段 28、ルート選択手段 29 が備えられている。

【0114】ハイパー構造管理手段 20 はグラフ構造を成している構造化データをメモリに保持して管理しており、この構造化データはノード、リンク、リンク間関係、選択属性、属性マッピングによって構成されている。ノードは構造化データを構成する最小単位であり、文書データを構造化した本実施例では、ノードは文書内容であるテキスト情報を保持している。なお、ノードをリンクで結合した木構造が、文書を表すカレント構造である。リンクは二つのノード間の結合関係を表す単位であり、リンクは始点となるノードと終点となるノードの組として定義される。なお、本実施例では、リンクはリンク間関係を記述する“LinkList”の記述に含まれる。また、このハイパー構造管理手段 20 は、請求項に言うノード保持手段、リンク保持手段及びリンク関係保持手段を構成している。

【0115】リンク間関係は同じノードを始点とするリンクの集合に対する関係を記述しており、この集合に含まれるリンクのリストと各リンクの位置を規定する。なお、本実施例では、リンク集合は順序を規定されている。選択属性は上記実施例の選択情報に対応した情報であり、同じノードを始点とする一つ以上のリンク間関係の中からリンク間関係を選択するための属性情報である。なお、本実施例では、作成日時、作成者名、バージョン名のリストを、選択属性として用いている。属性マッピングは選択属性をリンク間関係に関連付ける記述であり、本実施例では、リンク間関係を記述する“LinkList”が属性情報を保持する。

【0116】カレント構造管理手段 21 はノード、リンク、リンク間関係から成る木構造（グラフ構造の一種）で表現されるドキュメント構造データをメモリに保持して管理しており、このドキュメント構造が編集処理の対象となる。なお、本実施例では、リンク間関係およびリンクの記述はノードを記述する“Node”の記述に含まれる。編集手段 22 は、ユーザの指示に従ってドキュメント構造に含まれるノード、リンク、リンク間関係についての生成、更新、削除等といった編集処理を行い、候補表示手段 30、候補指示手段 31、交換手段 32 を有

している。候補表示手段30は同じノードを始点とするリンク間関係をディスプレイ画面に表示する手段であり、本実施例では、ユーザが選択したノードを始点とするリンク間関係を表示する。候補指示手段31は、ユーザの指示に従って候補表示手段30によって表示されたリンク間関係の中からいずれか一つを選択する手段である。交換手段32は、選択されているリンク間関係をユーザに指示されたリンク間関係に変更する手段である。

【0117】登録指示手段23はカレント構造管理手段21に保持されているドキュメント構造をハイパー構造管理手段20に登録させるための指示を、ユーザから受け取る手段であり、登録手段24は当該指示に従ってドキュメント構造をハイパー構造に登録する。なお、登録はカレント構造中のノード、リンク間関係をハイパー構造中に格納し、適切な属性情報マッピングを行うことである。履歴管理手段25はカレント構造の登録履歴を管理する手段であり、カレント構造管理手段21に保持されているドキュメント構造をハイパー構造管理手段20に登録させる毎に登録履歴を記録して管理する。なお、本実施例では、登録日時、登録者名、登録時に設定したバージョン名の組をセッションとして記録する。履歴表示手段26は履歴管理手段25で管理されているセッションの情報をディスプレイ画面に一覧として表示する手段である。

【0118】抽出手段27は、ユーザからの指示に従って選択されたリンク間関係を用いてハイパー構造管理手段20からドキュメント構造を抽出し、当該ドキュメント構造をカレント構造管理手段21へ保持させる手段であり、検索手段33と選択手段34とを有している。検索手段33は、同じノードを始点とするリンク間関係の中から、ユーザから指示された検索条件を満たすリンク間関係を検索する手段であり、選択手段34は、検索したリンク間関係の中から、ユーザから指示された選択規則に従って1つのリンク間関係を選択する手段である。

【0119】抽出指示手段28はユーザから抽出条件の指示を受け取る手段であり、検索条件指示手段35と選択規則指示手段36とを有している。なお、本実施例では、抽出条件としては選択条件と選択規則の二つが指示可能である。検索条件指示手段35はユーザから選択条件（すなわち、検索条件）の指示を受け取る手段であり、本実施例では、選択条件は日時、バージョン名、執筆者名を含んでいる。選択規則指示手段36はユーザから選択規則の指示を受け取る手段であり、本実施例では、選択規則はユーザ優先、日時優先、バージョン名優先のいずれかである。

【0120】ルート選択手段29は、カレント構造のルートとなるノードをハイパー構造管理手段20に保持されているハイパー構造の中から選択する手段であり、本実施例では、ルートノードを起点としてリンク間関係を順次選択することで、カレント構造管理手段21に保持さ

れるカレント構造を生成している。なお、上記した各機能手段20～36は、本実施例ではプロセッサが所定の制御プログラムを実行することにより構成されている。

【0121】ここで、上記した抽出指示手段28、履歴表示手段26、編集手段22は、ユーザに対して情報を提示し或いはユーザが指示や情報等を入力するためのインタフェース画面を表示出力する機能を有している。抽出指示手段28は図49に示すようなディスプレイ画面を表示出力し、この画面には次のような各機能部が設定されている。ルートノード名入力部40はハイパー構造管理手段20から抽出する文書データのルートとなるノード名を入力するためのエリアであり、検索条件入力部41は選択条件を入力するためのエリアである。この検索条件入力部41には上記した選択条件を入力するために、現在使用しているユーザの名前を入力するためのエリアであるユーザ名部42、抽出したい文書データの状態があった日時を入力するためのエリアである日時部43及び44、抽出したい文書のバージョン名を入力するためのエリアであるバージョン名部45が設けられている。

【0122】選択規則入力部46はユーザが選択規則のいずれかを指定するエリアであり、ユーザ優先（USER）を選択すると選択手段34が選択条件を満たすものから指定されたユーザに一致するものを優先的に選択する処理を行い、日時優先（DATE）を選択すると選択手段34が選択条件を満たすものから指定日時を基準として生成日時が最も新しいものを優先的に選択する処理を行い、バージョン名優先（VERSION）を選択すると選択手段34が選択条件を満たすものから指定されたバージョン名を含むものを優先的に選択する処理を行う。また、執筆者選択部47はユーザが選択条件の一部としてユーザ以外の執筆者を選択することができるエリアである。

【0123】また、操作ボタン部48には“extract”ボタンと“history”ボタンとが設けられている。“extract”ボタンを選択すると、指示されたルートノードを始点として選択条件と選択規則に基づいて文書データをハイパー構造管理手段20から抽出する処理が開始され、これと同時に履歴管理手段25が起動される。また、“history”ボタンを選択すると、履歴表示手段26が起動されて、それまでに行われた文書の登録履歴が画面に表示される。図50には、履歴表示手段26が画面表示する登録履歴の例を示してあり、一覧表示された各履歴（すなわち、各行）に一回の登録処理に関する情報として、日時、ユーザ名、バージョン名が表示される。

【0124】編集手段22は図51に示すようなディスプレイ画面を表示出力し、この画面には上記の“extract”ボタンの選択によって生成されて表示出力される。この画面には、文書構造編集ウィンドウ50、文

書内容編集ウィンドウ51、候補選択ウィンドウ52、編集ボタン53、抽出条件表示部54が設けられている。文書構造編集ウィンドウ50は抽出手段27によって抽出された文書データの構造を表示し、ユーザはポインティングデバイスによって文書構造に含まれるノードを選択することができる。

【0125】文書内容編集ウィンドウ51は、文書構造編集ウィンドウ50で選択されたノードの内容を表示し、ユーザは表示しているテキスト内容を編集することによって、選択されたノードの内容を更新することができる。候補選択ウィンドウ52は選択したノードを始点とするリンクリストの候補を表示し、現在選択されているリンクリストを反転して示す。このウィンドウ52において、ユーザはポインティングデバイスによって、他のリンクリストを選択して現在選択しているものと入れ換えることができる。

【0126】編集ボタン53には次の各ボタンが設けられており、ユーザはこれらボタンを用いて文書の編集と保存を行わせることができる。"add child" ボタンを選択すると現在選択されているノードの子ノードが生成され、"add brother" ボタンを選択すると現在選択しているノードの兄弟ノードが生成され、"move down" ボタンを選択すると現在選択しているノードが同じレベルでより後に移動され、"move up" ボタンを選択すると現在選択しているノードが同じレベルでより前に移動され、"move left" ボタンを選択すると現在選択しているノードのレベルが一段上げられ、"move right" ボタンを選択すると現在選択しているノードのレベルが一段下げられ、"move node" ボタンを選択すると現在選択しているノードが削除され、"save" ボタンを選択するとカレント構造管理手段21が保持しているドキュメントがハイパー構造管理手段20が保持しているハイパー構造中に保存され、"close" ボタンを選択すると全体のウィンドウを閉じられて編集処理が完了する。また、抽出条件表示部54にはカレント構造を抽出するための条件として指定された選択条件および選択規則が表示される。

【0127】次に、本実施例に係る構造化データの処理を、図52に示す例を用いてより具体的に説明する。まず、図52には、3名の執筆者、Peter、David、Wickyによる文書の編集活動の履歴を示してあり、図中の横軸は時間の経過を、縦軸は担当者を示している。また、同図中の(a)～(e)は、その縦軸の位置に対応する執筆者によって、横軸の位置に対応する日時において、ハイパー構造管理手段20へ保存(登録)された文書データの構造を示している。なお、図53は、編集の履歴を従来技術である導出木によって示したものであり、図中の矢印は導出関係を表しており、導出関係とは始点に位置する文書構造から終点に位置する文書構造が作られた

ことを意味する。このような導出関係は、導出木とよばれる各時点での文書の状態をノードとする木構造を構成する。

【0128】図52では文書構造の木構造はインデントされたテキストによって示しており、図54では(a)～(c)の表している木構造を表現したデータ構造を示している。図55～図57は、(a)～(c)の文書構造の保存が行われたときのハイパー構造の状態を示している。この例は、"WriteAtWill"という製品のマニュアルをPeter、David、Wickyの3人で執筆している例である。以下順を追って作業の経緯を説明する。

【0129】まず、(Aug 1)において、Peterが"WriteAtWill"のマニュアルを書き始め、この最初の状態である文書データ(a)を"PC version"という名前で保存した。次いで、(Aug 2)に、DOSとWINとの2つのOS用の記述が必要になったので、文書データ(a)の章立てをDOSとWINとに分けた文書データ(b)を作成し、名前を付けずに保存した。その後、2つの異なるマニュアルをそれぞれ別の担当者で執筆することが決まり、それぞれDavidとWickyとに担当が割り当てられた。そして、(Aug 3)に、Davidが文書データ(a)の中からDOS用の記述を取り出して加筆した文書データ(c)を作成し、"DOS version"という名前で保存した。更に、(Aug 4)に、Wickyが文書データ(b)の中からWIN用の記述を取り出して加筆した文書データ(e)を作成し、"WIN version"という名前で保存した。そして、(Aug 5)に、Davidが文書データ(c)にさらに加筆した文書データ(d)を作成し、名前を付けずに保存した。

【0130】このような場合において、本実施例では過去に存在していなかった文書データを生成することができる。すなわち、本発明は、従来技術のように過去に存在していた文書のバージョンが復元できるだけでなく、存在していなかった文書の構造を取り出せる。図58の文書データ(g)及び(f)は、図53に示した導出木上にはない文書データである。この文書データ(g)の構造は、Wickyの(Aug 4)の構造の内の"INTRO"以下の部分をDavidが(Aug 3)に加筆したものに置き換えて得られる構造である。すなわち、文書データ(g)は、Wickyにとっては自分の書いた部分はそのままにして、他の執筆者の書いた最新の情報を反映させたものである。また、文書データ(f)は、Peterが(Aug 2)に定義した全体の枠組みの中に、他の執筆者の書いた最新の情報を組み込んだものである。したがって、本発明では、このような過去に存在せず、また、予定もしていなかった新しい文書構造を、過去に作られた文書の部分を利用して簡単に取り出すことが可能となっている。

【0131】ここで、(Aug 6)の時点で文書を編集するときの選択条件と抽出結果のバリエーションを示

す図 5 9～図 7 4 を参照して、文書抽出の処理をより具体的に説明する。図 5 9 には履歴表示手段 2 6 で表示される (Aug 6) 以前に行われた文書保存の履歴を示してあり、図 6 0 には David によって指定された選択条件を示してある。図 6 0 に示す画面上において、ルートノード名入力部 4 0 にはルートノードの名前としてマニュアルのタイトルである "WAW Manual" が入力指定され、ユーザ名入力部 4 2 にはユーザの名前として "David" が入力指定され、日時入力部 4 3 にはいつまでに作成された内容を探索するかを示す日時の情報として日付 "Aug/06/96" 及び時間 "19:03" が入力指定され、バージョン名入力部 4 5 には検索を開始する起点となるバージョンを示すバージョン名として "DOS version" が入力指定されている。

【0132】なお、ユーザ以外のどの執筆者が作成した内容を探索するかを指定する執筆者選択部 4 7 には、この例では、誰も指定していないのでユーザである David が作成した文書内容が検索される。また、この例では、選択規則入力部 4 6 で "USER" が選択されており、自分で作成した内容を優先して選択することが指示されている。

【0133】図 6 1 には、図 6 0 に示した条件によってハイパー構造管理部 2 0 から文書データを抽出した結果を示してある。なお、この例では、図 5 2 に示した文書データ (d) が抽出されている。また、図 6 0 に示した条件を変更して、図 6 2 に示すように日時入力部 4 3 の条件を "Aug/03/96" に指定すると、(Aug 3) まですでで最新の内容を反映した文書データの抽出が指示される。この結果、図 6 3 に示すような文書構造がハイパー構造管理部 2 0 から抽出されて画面表示される。なお、この抽出された文書データは、この例では、図 5 2 に示した文書データ (c) である。

【0134】また、図 6 0 に示した条件を変更して、図 6 4 に示すようにユーザ入力部 4 2 の指定を "Wicky" とし、バージョン名入力部 4 5 の指定をバージョン名 "WIN version" とすると、図 6 5 に示すような文書構造がハイパー構造管理部 2 0 から抽出されて画面表示される。なお、この抽出された文書データは、この例では、図 5 2 に示した文書データ (e) である。また、図 6 4 に示した条件を変更して、図 6 6 に示すように選択規則入力部 4 6 の指定を "DATE" にし、さらに、執筆者選択部 4 7 の指定を他の執筆者として David と Wicky を加えるように指定すると、図 6 7 に示すような文書構造がハイパー構造管理部 2 0 から抽出されて画面表示される。なお、この抽出された文書データは、この例では、図 5 8 に示した文書データ (g) であり、図 5 2 に示した導出木の中には存在しない、従来技術では選択できなかった

データ構造が抽出されている。

【0135】また、図 6 8 に示すようにユーザ名入力部 4 2 の指定を "Peter" にし、さらにバージョン名入力部 4 1 の指定を "PC version" に変更すると、図 6 9 に示すような文書構造がハイパー構造管理部 2 0 から抽出されて画面表示される。なお、この抽出された文書データは、この例では、図 5 2 に示した文書データ (b) である。また、図 6 8 に示した条件を変更して、図 7 0 に示すように選択規則入力部 4 6 の指定を "VERSION" とすると、図 7 1 に示すような文書構造がハイパー構造管理部 2 0 から抽出されて画面表示される。なお、この抽出された文書データは、この例では、図 5 2 に示した文書データ (a) である。

【0136】また、図 6 8 に示した条件を変更して、図 7 2 に示すように、執筆者選択部 4 7 の指定を他の執筆者として David と Wicky を加えるように指定すると、図 7 3 に示すような文書構造がハイパー構造管理部 2 0 から抽出されて画面表示される。なお、この抽出された文書データは、この例では、図 5 8 に示した文書データ (g) であり、図 5 2 に示した導出木の中には存在しない、従来技術では選択できなかったデータ構造が抽出されている。ここで、図 7 3 に示す画面では、さらに最上位のレベルで入れ換え可能なリンクリストの候補が候補選択ウィンドウ 5 2 に表示され、また、現状の選択状況が反転表示されている。この例では、4 つの選択肢の中から Wicky の作成した、"Win version" のバージョン名のリンクリストが選択されていることを反転表示で示している。

【0137】また、図 7 4 には、図 7 3 において選択肢として示されているリンクリストから過去に Peter が作成したものを選択し、現状のものと入れ換える指示を行った状態を示してある。この結果、図 5 8 における文書データ (f) の構造が抽出され、図 5 2 に示した導出木の中には存在しない、従来技術では選択できなかったデータ構造が抽出される。

【0138】次に、本実施例のデータ構造とアルゴリズムについて説明する。本実施例は "Windows95" (登録商標) 上で、"Java" (登録商標) 言語によって実現されている。"Java" はオブジェクト指向言語であり、オブジェクトと呼ばれるデータ構造と、そのオブジェクトが受け付け実行するメソッドと呼ばれる手続きを、クラスの定義として記述する。以下に示すプログラムリストは、本実施例を実現している Java プログラムから主要な部分を抜粋、簡略化したものであり、このプログラムリストを参照しながらデータ構造とアルゴリズムを説明する。なお、データ構造は図 7 5 に示す。

【0139】

(1) リンク間関係記述の集合 "LinkedListHolder" のクラス定義

```
「1」 public class LinkedListHolder extends Object{
```

```
// LinkedListHolder は、同じノードを始点に持つリンクリストの集合を管理す
```

39

るオブジェクト。

/* データ構造の定義 */

「2」 Node sourceNode;

//始点となるノード

「3」 Vector linkListSet;

//リンクリストの集合

/* メソッドregisterLinksの定義 */

「4」 void registerLinks(Node node, String user, Date date, String version) {

//指定されたノードを始点とするリンクリストをリンクリストセットに登録(追加)するメソッド

【0140】

「5」 int index = -1;

//リンクリストセット中の同じリンクリストの番号を格納する変数

「6」 LinkList linkList;

//リンクリストセット中の同じリンクリストを格納する変数

「7」 for (int i=0; i<linkListSet.size();i++) {

//リンクリストセットに登録されている中から同じリンクリストを探す

「8」 linkList = (LinkList)(linkListSet.elementAt(i));

「9」 if (linkList.isLinksFor(node)) {

「10」 index = i;

「11」 break;

「12」 }

「13」 }

「14」 if (index >= 0) {

//同じリンクリストが見つかったとき

【0141】

「15」 if (version.length() == 0) {

//バージョン名が指定されたかをチェック

「16」 } else {

「17」 linkList.addVersion(version);

//見つかったリンクセットにバージョン名を追加

「18」 }

「19」 }else {

//同じリンクリストが見つからないとき

「20」 linkList = new LinkList(node, user, date, version);

//nodeのもつリンクリストにユーザ名、日時、バージョン名を対応付けたリンクリストを生成

【0142】

「21」 addLinkList(linkList);

//新たに生成したリンクリストを追加する。

「22」 }

「23」 }

/* メソッドselectLinksの定義 */

「24」 Vector selectLinks(String user, Vector authors, Date date, String version, int rule) {

//抽出条件を満たすリンクのリストから指定された規則(rule)にしたがって一つを選択するメソッド。LinkListSetは登録日時の新しい順に並ぶように登録しているので、最初に見つかったものが最も新しいリンクリストになる。

「25」 int userindex = -1;

41

//ユーザ名が合致する最初のリンクリストの番号を格納する変数

【0143】

```
「26」      int dateindex = -1;
            //日時の条件が合致する最初のリンクリストの番号を格納する変数
「27」      int versionindex = -1;
            //バージョン名が合致する最初のリンクリストの番号を格納する変数
「28」      for (int i=0; i<linkListSet.size(); i++) {
            //リンクリストセット中のすべてのリンクリストを探索
「29」          LinkList linkList = (LinkList)linkListSet.elementAt(i);
「30」          if (linkList.satisfies(authors, date, version)) {
            //執筆者、日時、バージョン名に関する条件を満たすものについて以下を実行
```

【0144】

```
「31」          if (dateindex < 0) {
            // まだ日時の条件を満たすものがみつかっていなければ
「32」              dateindex = i;
            // このリンクリストを最新のものとする
「33」          }
「34」          if (linkList.regUser.equals(user)) {
            // ユーザ名に合致するものがあれば
「35」              if (userindex < 0) {
            // まだユーザ名に合致するものがみつかっていなければ
```

【0145】

```
「36」              userindex = i;
            // このリンクリストを最新のものとする
「37」          }
「38」          }
「39」          if (linkList.hasVersion(version)) {
            // バージョン名が合致するものがあれば
「40」              if (versionindex < 0) {
            // バージョン名が合致するものがみつかっていなければ
```

【0146】

```
「41」              versionindex = i;
            // このリンクリストを最新のものとする
「42」          }
「43」          }
「44」          }
「45」      }
```

【0147】

```
「46」      int index = -1;
            // 選択するリンクリストの番号を格納する変数
「47」      if (rule == DATE) {
            // 選択規則が日時最新のと看
「48」          index = dateindex;
            // 日時が最新のものを選択
「49」      } else if (rule == USER) {
            // 選択規則がユーザ名優先のと看
「50」          if (userindex < 0) {
            // ユーザ名に合致するものがなければ
```

【0148】

```
「51」          index = dateindex;
```

43

// 最新のものを選択

```

「52」         } else {
// ユーザ名に合致するものがあれば
「53」         index = userindex;
// ユーザ名に合致する最新のものを選択する
「54」         }
「55」         } else if (rule == VERSION) {
// 選択規則がバージョン名優先のとき

```

【0149】

```

「56」         index = versionindex;
// バージョン名が合致する最新のものを選択する
「57」         }
「58」         if (index < 0) {
// もし選択されたものがなければ、
「59」         return new Vector();
// 空のリストを返す
「60」         } else {

```

【0150】

```

「61」         return ((LinkedList)linkListSet.elementAt(index)).links;
// 選択されたリンクリストを返す
「62」         }
「63」     }
「64」 }

```

【0151】

(2) ノード記述Nodeのクラス定義

```

「65」 public class Node extends Object {
// ノードを記述するオブジェクト
/* データ構造 */
「66」     String contents;
// ノードの保持するテキスト内容
「67」     Vector links;
// ノードを始点とするリンクのリスト
「68」     Long nodeID;
// ノードの識別子。この値が一致するノードを同一のノードと見なす。
「69」 }

```

【0152】

(3) リンク間関係記述LinkedListのクラス定義

```

「70」 public class LinkedList extends Object {
// 同じノードを始点とするリンクのリストを表すオブジェクト
/* データ構造の定義 */
「71」     String sourceNodeContents;
// ソースノードの内容
「72」     Vector links;
// 終点ノードのリスト
「73」     Date regDate;
// 作成日時
「74」     String regUser;
// 作成者
「75」     Vector versions;
// バージョン名のリスト

```

45

【0153】

```

/* メソッドisLinksForの定義 */
「76」    public boolean isLinksFor(Node node) {
           // 指定されたノードのリンクリストに一致するかを調べるメソッド。
「77」        if (!sourceNodeContents.equals(node.contents)) return false;
           // ソースノードと内容が違えば一致していない
「78」        if (links.size() != node.links.size()) return false;
           // リンクリストの長さが違えば一致しない
「79」        boolean equal = true;
           // 一致しないものが一つでもあるかどうかを調べる
「80」        for (int i=0; i < links.size(); i++) {

```

【0154】

```

「81」            equal = ((Node)links.elementAt(i)).equals((Node)node.links.elementAt(i));
「82」            if (!equal) break;
「83」        }
「84」        return equal;
「85」    }

```

【0155】

```

/* メソッドsatisfiesの定義 */
「86」    blic boolean satisfies(Vector authors, Date date, String version)
    {
        // 条件を満たすかどうかを調べるメソッド
「87」        if (regDate.after(date)) {
            // 登録日時が指定された日時よりも後であれば満たさない
「88」            return false;
「89」        }

```

【0156】

```

「90」        for (int i=0; i<versions.size(); i++) {
            // 指定されたバージョン名を含んでいれば満たす
「91」            if (((String)versions.elementAt(i)).equals(version)) {
「92」                return true;
「93」            }
「94」        }
「95」        for (int i = 0; i < authors.size(); i++) {
            // 指定された執筆者を含んでいれば満たす。

```

【0157】

```

「96」            String au = (String)authors.elementAt(i);
「97」            if (regUser.equals(au)) {
「98」                return true;
「99」            }
「100」        }
「101」        return false;
「102」    }
「103」}

```

【0158】

```

(4) ハイパー構造管理手段HyperStructureManagerのクラス定義
「104」 public class HyperStructureManager extends Object {
           // ノードとリンクからなる全体構造を管理するオブジェクト
/* データ構造の定義 */

```

47

```

「105」    Vector linkListHolderSet;
           // 含まれるリンクリストの集合
「106」    Vector regVersions;
           // 登録されているバージョン名のリスト
「107」    Vector regAuthors;
           // 登録されている著者名のリスト
「108」    Vector sessionHistory;
           // 実行された登録操作の履歴

```

【0159】

```

/* メソッドsaveDocumentの定義 */
「109」    void saveDocument(Node root, String user, Date date, String ve
rsion) {
           // ルートノードによって代表されるカレント構造を登録するメソッド
「110」        if (root == null) {
「111」            } else if (regVersionsHas(version)) {
           // 既にバージョン名が登録されていればなにもしない。
「112」            } else {
「113」                if (version.length() > 0) {
           // バージョン名が指定されていれば
「114」                    regVersions.addElement(version);
           // 指示されたバージョン名を登録する
「115」                }

```

【0160】

```

「116」        saveTree(root, user, date, version);
           // ルートノードからトラバースできる構造をハイパー構造に登録する
「117」        sessionHistory.addElement(new Session(user, date, vers
ion));
           // 登録操作の履歴を追加
「118」    }
「119」    }

```

【0161】

```

/* メソッドsaveTreeの定義 */
「120」    void saveTree(Node node, String user, Date date, String versio
n) {
           // 指示されたノードからトラバースできる構造をハイパー構造に登録する
           メソッド
「121」        int index = -1;
           // ハイパー構造からノードを始点とするリンクリストホルダーを探す
「122」        for(int i=0; i<linkListHolderSet.size();i++) {
「123」            if (((LinkListHolder)linkListHolderSet.elementAt(i)).s
ourceNode.equals(node)) {
「124」                index = i;
「125」                break;

```

【0162】

```

「126」        }
「127」    }
「128」    LinkListHolder llholder;
           // 指示されたノードを始点とするリンクリストホルダーを格納する変数
「129」    if(index < 0) {
           // リンクリストホルダーが見つからなければ

```

49

```
「130」          llholder = new LinkListHolder(node);
                // 新しいリンクリストホルダーを生成して選択する
```

【0163】

```
「131」          linkListHolderSet.addElement(llholder);
                // 生成したリンクリストホルダーを追加する
「132」          } else {
「133」          llholder = (LinkListHolder)linkListHolderSet.elementAt
(index);
                // 見つければそのリンクリストホルダーを選択する。
「134」          }
「135」          llholder.registerLinks(node, user, date, version);
                // リンクリストホルダーに指示されたノードのリンクリストを追加する
```

【0164】

```
「136」          for (int i = 0; i < node.links.size(); i++) {
「137」          saveTree((Node)node.links.elementAt(i), user, date, versi
on);
                // 指示されたノードのリンクの終点のノードについてsaveTreeメソッドを
再帰的に呼び出す
「138」          }
「139」          }
```

【0165】

```
/* メソッドextractDocumentの定義 */
「140」          void extractDocument(Node root, String user, Vector authors, D
ate date, String version, int rule) {
                // ルートノードとして指定されたノードを起点にして文書を表すカレント
構造を抽出するメソッド
「141」          LinkListHolder linkListHolder = linkListHolderFor(root);
                // ルートノードを始点とするリンクリストホルダーを選択する
「142」          root.contents = linkListHolder.sourceNode.contents;
                // ルートノードの内容としてリンクリストホルダーの内容を読み込む
「143」          Date sesdate = sessionDateOf(version);
                // バージョン名が登録された日時を取り出す
「144」          if (sesdate != null) {
「145」          if (date.before(sesdate)) {
                // 指定された日時がバージョン名の登録日より以前かどうかを調べる
```

【0166】

```
「146」          } else {
「147」          extractTree(root, user, authors, date, version, ru
le);
                // 登録日以降の指定日のときにルートノードを起点として構造を抽出する
「148」          }
「149」          }
「150」          }
```

【0167】

```
/* メソッドextractTreeの定義 */
「151」          void extractTree(Node node, String user, Vector authors, Date
date, String version, int rule) {
                // 指定されたノードを起点として構造を抽出するメソッド
「152」          LinkListHolder llholder = linkListHolderFor(node);
                // 指定されたノードを始点とするリンクリストホルダーを選択する
```

51

52

```

「153」      Vector links = llholder.selectLinks(user, authors, date, v
ersion, rule);
// リンクリストホルダーの中から抽出条件を満たすリンクを選択する
「154」      node.links = new Vector();
// ノードのリンクリストを初期化
「155」      Node nd = null;

【0168】

「156」      for (int i=0; i < links.size(); i++) {
// 選択されたリンクリストの各リンクについて以下を実行
「157」          nd = ((Node)links.elementAt(i)).copy();
// リンクの終点ノードをコピーして新しいノードを生成する
「158」          node.addNode(nd);
// 生成したノードを指定されたノードのリンクリストに追加する
「159」          extractTree(nd, user, authors, date, version, rule);
// 生成したノードを起点にしてextractTreeを再帰的に実行する
「160」      }
「161」    }

```

【0169】また、図75には、本実施例の主要な構成要素のデータ構造を示してある。同図(a)に示すカレント構造は、本実施例では、編集対象の文書表現するデータ構造であり、このカレント構造をノード記述“Node”からなる木構造の形で表現している。この“Node”のデータ構造は上記したリストの65行目から68行目までが対応するオブジェクトの定義であり、次の記述を含んでいる。“contents”は文書内容を有しており、本実施例では文字列である。“links”は1つ以上の“Node”への参照のリストを有しており、このリストはこの“Node”を始点とするリンクの集合と順序関係を定めるリンク間関係記述である。“nodeID”はノードの識別子を有しており、この値によってノードが同一であるかどうか判断される。

【0170】図75(b)に示すハイパー構造は、ハイパー構造管理手段20で保持管理される構造化データであり、本実施例では、このハイパー構造を“LinkedList”、“LinkedListHolder”によって構造化された“Node”のネットワーク構造の形で表現している。文書データの構造は、“Node”の“links”による他の“Node”への参照を用いて構成されるが、ハイパー構造中では“LinkedList”と“LinkedListHolder”からの参照によって構成される。このハイパー構造は次の記述を含んでいる。“LinkedList”は、同じノードを始点とするリンクの集合とその間の関係を記述するリンク間関係記述であり、さらに、このリンク間関係記述に対応づける選択属性も管理する。なお、上記したリストの70行目から75行目までが対応するオブジェクトの定義である。

【0171】また、“sourceNodeContents”はリンクの始点となるノードの文書内容を有しており、“links”はリンクの終点となるNodeへの参照のリストを有している。また、“regDate”は選択属性である作成日時を有し、“regUser”は選択属性である作成者名を有し、“v

ersions”は選択属性であるバージョン名のリストを有している。“LinkedListHolder”は、同じNodeを始点とするLinkedListの集合を保持しており、上記したリストの1行目から3行目までが対応するオブジェクトの定義である。また、“sourceNode”は始点となるNodeを参照し、“linkListSet”は同じNodeを始点とするLinkedListの集合を有している。

【0172】ハイパー構造管理手段20が保持管理するデータ構造は、図75(c)に示してあり、本実施例では、ハイパー構造管理手段は“HyperStructureManager”というオブジェクトとして実現されている。“HyperStructureManager”は以下の情報を管理しており、上記したリストの65行目から68行目までが対応するオブジェクトの定義である。“LinkedListHolderSet”は“LinkedListHolder”の集合を有し、“regVersions”はカレント構造の登録時に登録されたすべてのバージョン名のリストを有し、“regAuthors”はこの構造を編集できる執筆者の名前のリストを有し、“sessionHistory”はこの構造に対して行った文書登録操作の履歴情報を有している。

【0173】次に、上記したリストを参照して本実施例のアルゴリズムを説明する。“HyperStructureManager”のメソッドは、上記リストの140行目から150目に対応する“extractDocument”、151行目から161行目に対応する“extractTree”、109行目から119行目に対応する“saveDocument”、120行目から138行目に対応する“saveTree”を有している。“extractDocument”はルートとして指示されたノードを起点にして、カレント構造管理手段21に保持するカレント構造をハイパー構造管理手段20に保持されたハイパー構造から抽出する処理を行う。この抽出処理では、ルートから開始して、下位に位置するノードについて下位構造を順次抽出するメソッド“extractTree”を呼び

出すことによって木構造を抽出する。

【0174】また、“extractTree”は、指定されたノードを起点としてデータ構造を抽出する処理を行い、指定されたノードを始点とするリンクリストホルダーから抽出条件を満たすリンクリストを選択し、その内容を持つノードを生成し、さらにその参照先のノードについて“extractTree”を再帰的に実行する。“saveDocument”は、ルートノードによって代表されるカレント構造をハイパー構造に登録する処理を行い、ルートノードが参照する下位のノードについて、下位構造に登録するメソッド“saveTree”を呼び出すことによって、木構造全体に登録する。“saveTree”は、指定されたノードの下位構造をハイパー構造に登録する処理を行い、指定されたノードを始点とするリンクリストホルダーにノードが保持するリンクリストに一致するものがなければ、リンクリストを追加し、さらにその下位ノードについて“saveTree”を再帰的に実行する。

【0175】次に、“LinkListHolder”のメソッドは、上記リストの4行目から23行目に対応する“registerLinks”、24行目から64行目に対応する“selectLinks”を有している。“registerLinks”は、指定されたカレント構造中のノードが持つリンクのリストを追加する処理を行い、追加対象のリンクリストと一致するものがすでに登録されているかを調べ、登録されていないければ、新しいリンクリストとして追加する。なお、すでに登録されていて、かつ、バージョン名が指定されているときには、バージョン名の設定を行う。“selectLinks”は、保持しているリンクリストの中から選択条件を満たすリンクリストをすべて検索し、その中から選択規則に従ってリンクリストを1つ選択する処理を行う。

【0176】次に、“LinkList”のメソッドは、上記したリストの76行目から85行目に対応する“isLinksFor”と、86行目から103行目に対応する“satisfies”を有している。“isLinksFor”は、カレント構造中のノードの持つリンクのリストと一致するかどうかを判断する処理を行い、ノードの文書内容と順序も含めたリンクの集合が一致するときに“True”を返す。“satisfies”は、選択条件を満たすかどうかを判断する処理を行い、指定された日時を含んでそれ以前に作成されたもので、指定されたバージョン名を含んでいるか、または、指定された執筆者によって作成されたものであれば“True”を返す。

【0177】上記のアルゴリズムにより、抽出指示手段28から指定された選択条件及び選択規則に従って、該当する文書構造が抽出手段27によってハイパー構造管理手段20から抽出され、抽出された文書構造が編集手段22による編集処理の対象としてカレント管理手段21に保持される。また、編集処理がなされた文書構造は登録手段24によってハイパー構造管理手段20へ登録され、この登録処理に際して履歴情報が履歴管理手段2

5によって登録される。

【0178】次に、本発明の構造化データ処理装置を図形の編集に適用した他の一実施例を説明する。なお、本実施例の構成、データ構造、アルゴリズムは、本質的に上記した構造化データ処理装置の実施例と同じであるので、重複する説明は割愛する。図76には、図形の編集過程の一例を示してあり、横軸に日時を、縦軸に執筆者名を示し、図中の(a)(b)(c)で示す図形が、縦軸の位置に対応する執筆者による編集結果として横軸の位置に対応する日時にハイパー構造管理手段20に保存されたことを示している。また、この例では、(Aug 1)にPeterが(a)に示す図形を作成し、保存した。そして、(Aug 2)にDavidが(b)に示すように、(a)に含まれる図形要素である文字列“蓄積部”の部分をデータベースの概念図を含むように変更し、保存した。さらに、(Aug 3)にPeterが(c)に示すように、(a)に新しい図形要素である文字列“計算部”を加えて保存した。

【0179】図77には、上記の(a)(b)(c)それぞれのデータ構造を示してあり、上記した実施例と異なる点は、文書内容として図形要素を持ち、リンク間関係が図形要素中の参照位置として二次元位置を規定していることである。それぞれの図(a)(b)(c)で、(1)はデータの論理的な構造を図示したものであり、(2)は文書内容であると同時にリンク間の関係を記述するために使われている図形要素との関係を含めて図示したものである。また、図78の(a)には、ハイパー構造管理手段20に保持されるハイパー構造中のデータ構造を示してある。また、同図の(b)には選択条件を示してあり、当該選択条件を指示することによって、図79の(a)(b)(c)に示す図形がカレント構造として抽出されてカレント構造管理手段21に保持される。なお、ここに示す抽出されたデータ構造は、PeterとDavidの両者の編集の最新状態を取り込んだものであり、過去には存在しなかった構造である。

【0180】次に、本発明の構造化データ処理装置をハイパーテキストの検索に適用した更に他の一実施例を説明する。なお、本実施例では、編集ではなく、検索履歴として生成される木構造を処理の対象にする点が異なるが、本質的なデータ構造やアルゴリズムは上記した実施例と同じである。

【0181】図80には、ある調査のために、Peterが(Aug 1)に行ったネットワーク検索の一例を示してある。すなわち、まず、http://AAA.htmlというアドレスのページを開いて、そこから関連するページBBBへのアンカーを選択してジャンプした。この結果、http://BBB.htmlというアドレスのページが開き、さらに、そこから関連するページDDDへのアンカーを選択してジャンプした。この結果、http://DDD.htmlというアドレスのページが開き、さらに、そこから前のページに戻る指

示をした。この結果、http://BBB.htmlというアドレスのページが開き、さらに、そこから関連するページEEEへのアンカーを選択してジャンプし、その結果として、http://EEE.htmlというアドレスのページが開いた。

【0182】図81には、同じ目的の調査のために、Davidが(Aug 2)に行ったネットワーク検索の一例を示してある。すなわち、まず、http://AAA.htmlというアドレスのページを開いて、そこから関連するページBBBへのアンカーを選択してジャンプした。この結果、http://DDD.htmlというアドレスのページが開き、さらに、そこから前のページに戻る指示をした。この結果、http://BBB.htmlというアドレスのページが開き、さらに、そこから関連するページCCCへのアンカーを選択してジャンプした。この結果、http://CCC.htmlというアドレスのページが開き、さらに、そこから関連するページFFFへのアンカーを選択してジャンプした。この結果、http://FFF.htmlというアドレスのページが開き、さらに、そこから前のページに戻る指示をした。この結果、http://CCC.htmlというアドレスのページが開き、さらに、そこから関連するページGGGへのアンカーを選択してジャンプし、この結果として、http://GGG.htmlというアドレスのページが開いた。

【0183】図82の(a)(b)には、上記したそれぞれの検索の履歴を木構造として表現してある。検索を開始したノードをルートにして、検索に利用したリンクだけを選択することによって、検索したデータ空間を表す木構造が生成される。なお、ハイパーテキストの検索履歴からこのような木構造を得るアルゴリズムとしては、特開平6-259312号公報に記載されている従来技術を採用している。なお、同図の(c)には、検索の履歴の構造をカレント構造として、ハイパー構造に登録した結果を示してある。なお、本実施例では、ハイパー構造は、WWWなどの分散したネットワーク構造の複数の部分構造を組み合わせて得られるものである。ハイパー構造は、検索を実行するクライアントのための情報として、ローカルに保持されるようにしてもよいし、ネットワークを維持している無数のサーバに分散して保持されるようにしてもよい。

【0184】図83の(a)には、ハイパー構造管理手段20からカレント構造を抽出するために指定した選択条件を示してあり、同図の(b)には、抽出されたカレント構造を示してある。この抽出されたカレント構造は、PeterとDavidの行った調査をマージした調査全体の検索空間を表わす過去に存在しなかった構造である。

【0185】なお、本発明は、上記した実施例以外に、例えば、部品のバージョンと全体構造のバージョンの両方を管理し、柔軟に組み合わせることに意味のあるような様々なアプリケーションに適用することもできる。また、本発明を適用することによって、それぞれの視点で書かれた新しい情報を制約なく反映させることができ、

また、過去の情報構造の断片を生かして新しい状況に合わせた構造を生成することができる。例えば、分散した環境での電子メールや電子掲示板での意見交換に適用することによって、過去の議論の復元や、過去の議論の断片から新しい情報の構造を構築することができる。

【0186】また、書いた人がお互いを知らないほど多数の文書を管理するシステムに適用することによって、コメントや修正によって直接文面を修正しても支障が起らず、さらに、新しい文書を合成する機能を持つ文書データベースを実現することができる。また、電子メールなどの流れの構造や組織の構造などの管理に本発明を適用することによって、過去の仕事の依頼の流れの復元や、それを用いた業務の分析や設計を行う機能を実現することができる。また、担当者が複数いるときばかりでなく、一人で文書を編集するシステムに適用した場合でも、当初想定していなかった目的のために、それまでに作ってきた文書から適切な部分を抜き出して必要な文書に再構成する機能を実現することができる。

【0187】

【発明の効果】以上説明したように、本発明に係る構造化データ処理装置によると、ノード及びリンクから構成されたグラフ構造の構造化データについて、リンク間関係保持手段に同一のノードを始点とする複数のリンクの内の同一の編集処理に係るリンクを集合として保持させるとともに各集合中に含まれるリンク間の位置関係情報を保持させるようにしたため、文書データ等のグラフ構造をなす構造化データについて、利用者に自由に編集をさせても、過去のデータ構造の復元や新しい状況に合わせた適切なデータ構造の作成を実現することができる。

また、本発明は、利用者による簡易な指定操作によって、過去に作ってきたデータを元にして、当初想定していなかった新しい状況に合わせた適切な構造化データを作り出すことを実現することができる。

【0188】特に、本発明に係る文書処理装置及び方法によると、論理単位を含む論理単位グループに階層構造によって文書構造を構成し、編集処理に際しては新たな論理単位グループを生成して元の論理単位グループと同一の集合として保持するようにしたため、文書の全体構造を編集によって変更した場合であっても、編集前の構造で文書を復元させることができる。したがって、執筆者に編集上の制約を与えずに文書を自由に編集させても過去の文書を支障無く復元することができ、共同執筆を支障無く実現させることができる。また、本発明に係る文書処理装置及び方法によると、各論理単位グループに選択情報を対応付けるようにしたため、利用者が選択情報を含む選択条件を指定するといった簡易な操作によって、適合する過去の文書バージョンを復元させることができ、任意のバージョンの文書について参照や編集処理を容易に行うことができる。

【図面の簡単な説明】

【図 1】 本発明の一実施例に係る文書処理装置の構成図である。

【図 2】 ストリームセットの構成を示す概念図である。

【図 3】 文書構造の一例を示す概念図である。

【図 4】 内容部が変更された文書構造の一例を示す概念図である。

【図 5】 部分構造が挿入された文書構造の一例を示す概念図である。

【図 6】 部分構造が削除された文書構造の一例を示す概念図である。

【図 7】 部分構造の順所が入れ替えられた文書構造の一例を示す概念図である。

【図 8】 部分構造が移動された文書構造の一例を示す概念図である。

【図 9】 ストリーム選択の原理を説明する概念図である。

【図 10】 文書の編集セッションの一例を示す図である。

【図 11】 セッション 1 において起動された文書エディタを示す図である。

【図 12】 セッション 1 において編集作業中の文書エディタを示す図である。

【図 13】 セッション 1 において生成された文書構造の一例を示す図である。

【図 14】 セッション 1 において保存処理のツールを示す図である。

【図 15】 セッション 1 においてメモリに格納された文書構造の一例を示す図である。

【図 16】 セッション 2 において文書フォルダ及び抽出条件設定ツールを示す図である。

【図 17】 セッション 2 において編集作業中の文書エディタを示す図である。

【図 18】 セッション 2 において生成された文書構造の一例を示す図である。

【図 19】 セッション 2 においてメモリに格納された文書構造の一例を示す図である。

【図 20】 セッション 3 において編集作業中の文書エディタを示す図である。

【図 21】 セッション 3 において生成された文書構造の一例を示す図である。

【図 22】 セッション 3 においてメモリに格納された文書構造の一例を示す図である。

【図 23】 セッション 4 において起動された抽出条件設定ツールを示す図である。

【図 24】 セッション 4 において編集作業中の文書エディタを示す図である。

【図 25】 セッション 4 において生成された文書構造の一例を示す図である。

【図 26】 セッション 4 において保存処理のツールを

示す図である。

【図 27】 セッション 4 においてメモリに格納された文書構造の一例を示す図である。

【図 28】 セッション 5 において編集作業中の文書エディタを示す図である。

【図 29】 セッション 5 において生成された文書構造の一例を示す図である。

【図 30】 セッション 5 においてメモリに格納された文書構造の一例を示す図である。

【図 31】 セッション 6 において起動された抽出条件設定ツールを示す図である。

【図 32】 セッション 6 において参照作業中の文書エディタを示す図である。

【図 33】 セッション 6 において生成された文書構造の一例を示す図である。

【図 34】 セッション 7 において編集作業中の文書エディタを示す図である。

【図 35】 セッション 7 において保存処理のツールを示す図である。

【図 36】 セッション 7 においてメモリに格納された文書構造の一例を示す図である。

【図 37】 文書抽出表示処理の手順の一例を示すフローチャートである。

【図 38】 下位構造抽出表示処理の手順の一例を示すフローチャートである。

【図 39】 ストリーム表示処理の手順の一例を示すフローチャートである。

【図 40】 ストリーム選択処理の手順の一例を示すフローチャートである。

【図 41】 文書再選択表示処理の手順の一例を示すフローチャートである。

【図 42】 文書保存処理の手順の一例を示すフローチャートである。

【図 43】 文書構造の表示形態の他の一例を示す図である。

【図 44】 文書構造の表示形態の更に他の一例を示す図である。

【図 45】 本発明の構造化データ処理に係るデータ構造のバリエーションを示す概念図である。

【図 46】 本発明の構造化データ処理に係るデータ構造のバリエーションを示す概念図である。

【図 47】 本発明の構造化データ処理に係るデータ構造のバリエーションを示す概念図である。

【図 48】 本発明に係る構造化データ処理装置の一実施例を示す構成図である。

【図 49】 抽出指示手段の画面表示の一例を示す図である。

【図 50】 履歴表示手段の画面表示の一例を示す図である。

【図 51】 編集手段の画面表示の一例を示す図であ

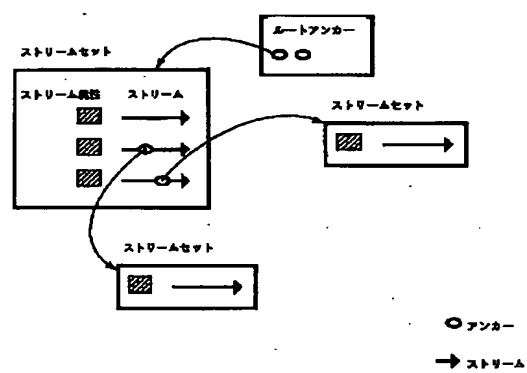
る。

- 【図 5 2】 編集履歴の一例を示す図である。
- 【図 5 3】 導出木構造の一例を示す図である。
- 【図 5 4】 編集中の文書データ構造の一例を示す図である。
- 【図 5 5】 ハイパー構造のデータ構造の一例を示す図である。
- 【図 5 6】 ハイパー構造のデータ構造の一例を示す図である。
- 【図 5 7】 ハイパー構造のデータ構造の一例を示す図である。
- 【図 5 8】 新たに生成される文書データの一例を示す図である。
- 【図 5 9】 履歴表示手段の画面表示の一例を示す図である。
- 【図 6 0】 抽出指示手段の画面表示の一例を示す図である。
- 【図 6 1】 編集手段の画面表示の一例を示す図である。
- 【図 6 2】 抽出指示手段の画面表示の一例を示す図である。
- 【図 6 3】 編集手段の画面表示の一例を示す図である。
- 【図 6 4】 抽出指示手段の画面表示の一例を示す図である。
- 【図 6 5】 編集手段の画面表示の一例を示す図である。
- 【図 6 6】 抽出指示手段の画面表示の一例を示す図である。
- 【図 6 7】 編集手段の画面表示の一例を示す図である。
- 【図 6 8】 抽出指示手段の画面表示の一例を示す図である。
- 【図 6 9】 編集手段の画面表示の一例を示す図である。
- 【図 7 0】 抽出指示手段の画面表示の一例を示す図である。
- 【図 7 1】 編集手段の画面表示の一例を示す図である。
- 【図 7 2】 抽出指示手段の画面表示の一例を示す図である。
- 【図 7 3】 編集手段の画面表示の一例を示す図である。
- 【図 7 4】 編集手段の画面表示の一例を示す図である。
- 【図 7 5】 カレント構造及びハイパー構造のデータ構

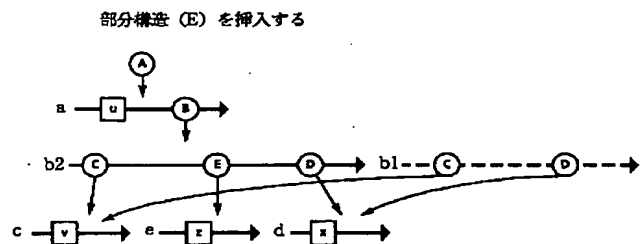
造の一例を示す図である。

- 【図 7 6】 本発明の他の一実施例に係る図形の編集過程を示す図である。
- 【図 7 7】 図形編集によるデータ構造の一例を示す図である。
- 【図 7 8】 ハイパー構造のデータ構造の一例を示す図である。
- 【図 7 9】 抽出された図形データ構造の一例を示す図である。
- 【図 8 0】 本発明の更に他の一実施例に係るハイパーテキストの編集過程の一例を示す図である。
- 【図 8 1】 本発明の更に他の一実施例に係るハイパーテキストの編集過程の他の一例を示す図である。
- 【図 8 2】 ハイパーテキストの履歴の一例を示す図である。
- 【図 8 3】 ハイパーテキストの抽出処理を説明する図である。
- 【図 8 4】 従来例に係る文書構造を示す概念図である。
- 【図 8 5】 内容部が変更された従来例の文書構造を示す概念図である。
- 【図 8 6】 部分構造が挿入された従来例の文書構造を示す概念図である。
- 【図 8 7】 部分構造が削除された従来例の文書構造を示す概念図である。
- 【図 8 8】 部分構造の順所が入れ替えられた従来例の文書構造を示す概念図である。
- 【図 8 9】 部分構造が移動された従来例の文書構造を示す概念図である。
- 【符号の説明】
- 1・・・メモリ、 2・・・論理単位保持手段、 3・・・論理単位グループ保持手段、 4・・・選択情報対応付け手段、 5・・・文書編集手段、 6・・・論理単位グループ生成手段、 7・・・参照関連付手段、 8・・・条件指定手段、 9・・・論理単位グループ選択手段、 12・・・文書構成手段、 13・・・文書構造表示手段、 14・・・ディスプレイ、 15・・・未表示情報表示手段、 16・・・論理単位グループ指定手段、 17・・・論理単位グループ再選択手段、 20・・・ハイパー構造管理手段、 21・・・カレント構造管理手段、 22・・・編集手段、 24・・・登録手段、 25・・・履歴管理手段、 27・・・抽出手段、 28・・・抽出指示手段、 33・・・検索手段、 34・・・選択手段、 35・・・検索条件指示手段、 36・・・選択規則指示手段、

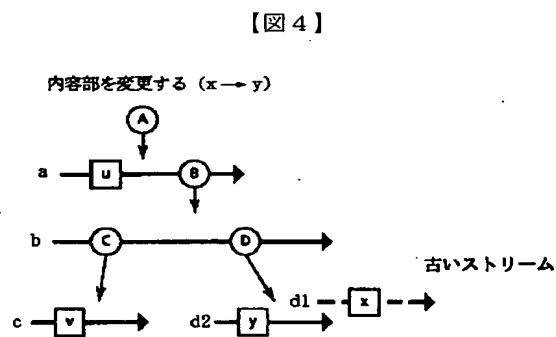
【図 2】



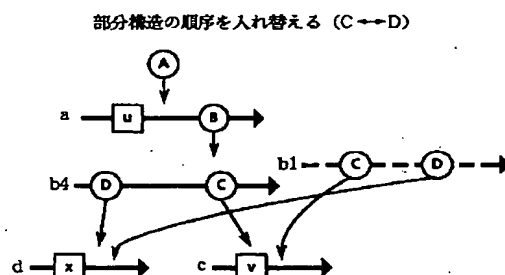
【図 5】



【图 1 1】



【図 7】



新編文書の名前を入力して下さい

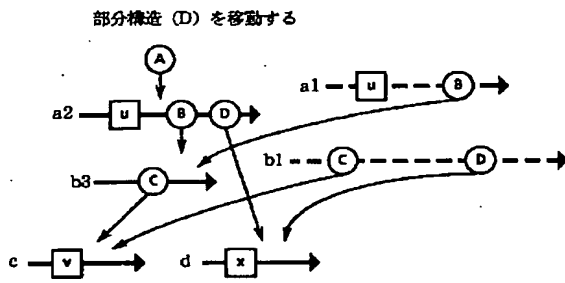
文書ルート: 免明説明書

ok

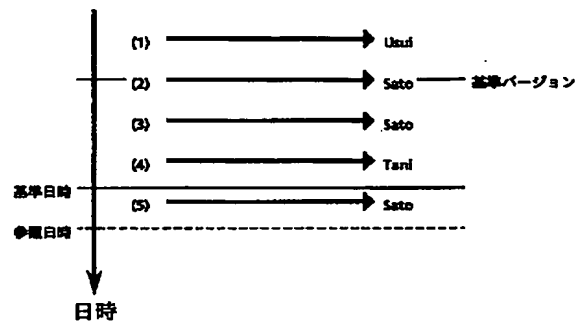
日時: 1995.1.1 15:00

ユーザ: Sato

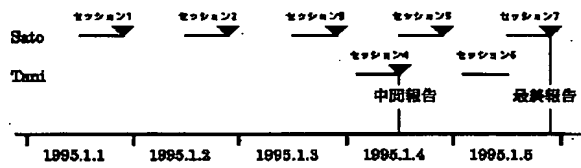
【図8】



【図9】

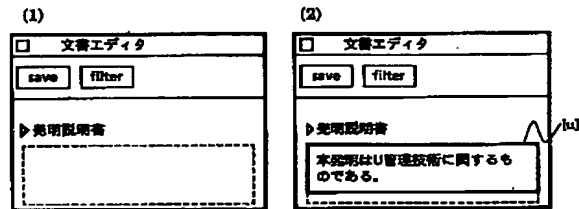


【図10】



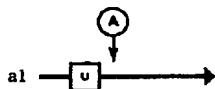
【図13】

【図12】



文書構造

【図14】



必要な設定を下下さい

文書ルート: 発明説明書

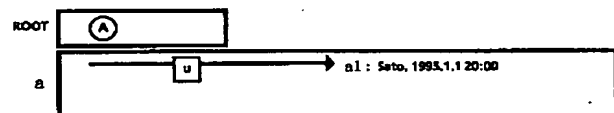
ok

バージョン設定:

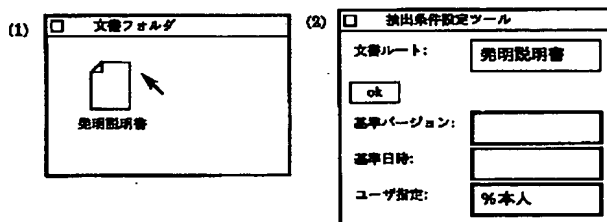
日時: 1995.1.1 20:00

ユーザ: Sato

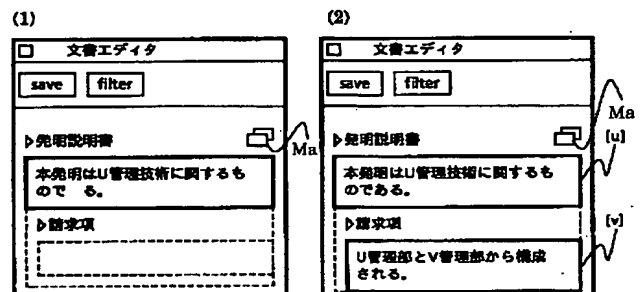
【図15】



【図16】

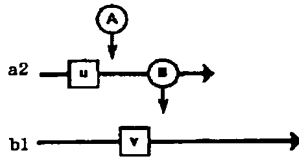


【図17】



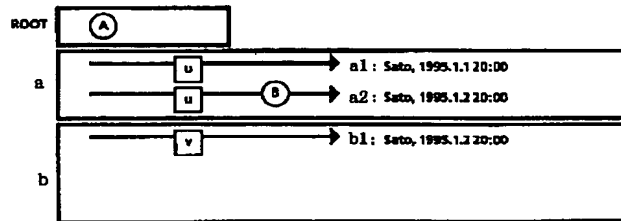
【図18】

文書構造



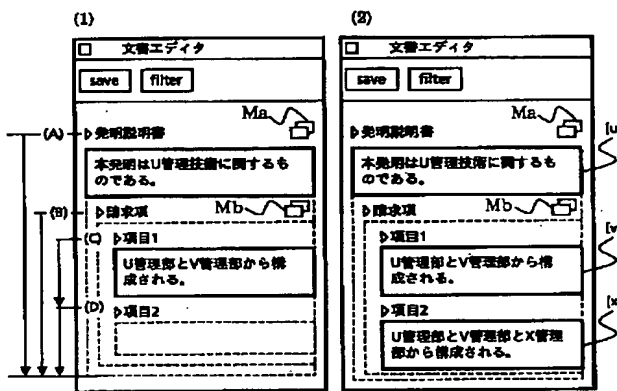
【図19】

複合文書構造

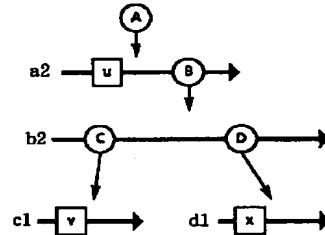


【図20】

【図21】



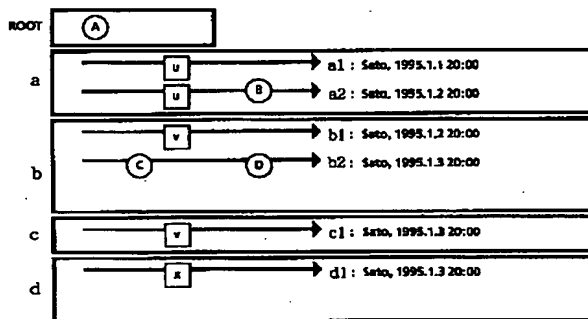
文書構造



【図22】

【図23】

複合文書構造



抽出条件設定ツール

文書ルート: 発明説明書

ok

基準バージョン:

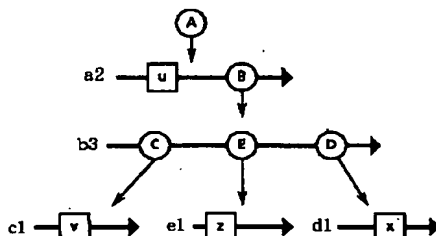
基準日時:

ユーザ指定:

【図35】

【図25】

文書構造



【図26】

必要な設定を下下さい

文書ルート: 発明説明書

ok

バージョン設定: 中間報告

日時: 1995.1.4 15:00

ユーザ: Tani

必要な設定を下下さい

文書ルート: 発明説明書

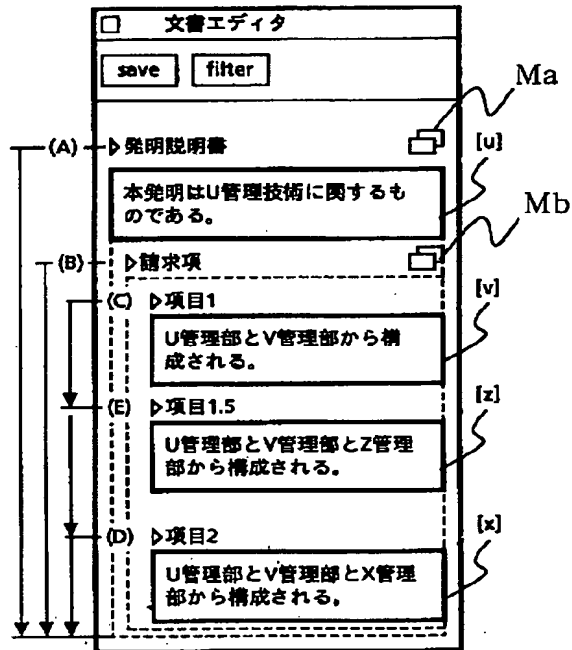
ok

バージョン設定: 最終報告

日時: 1995.1.4 20:00

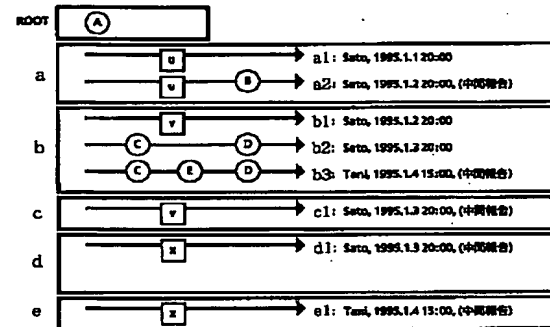
ユーザ: Sato

【図24】

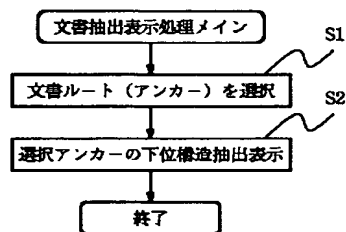


【図27】

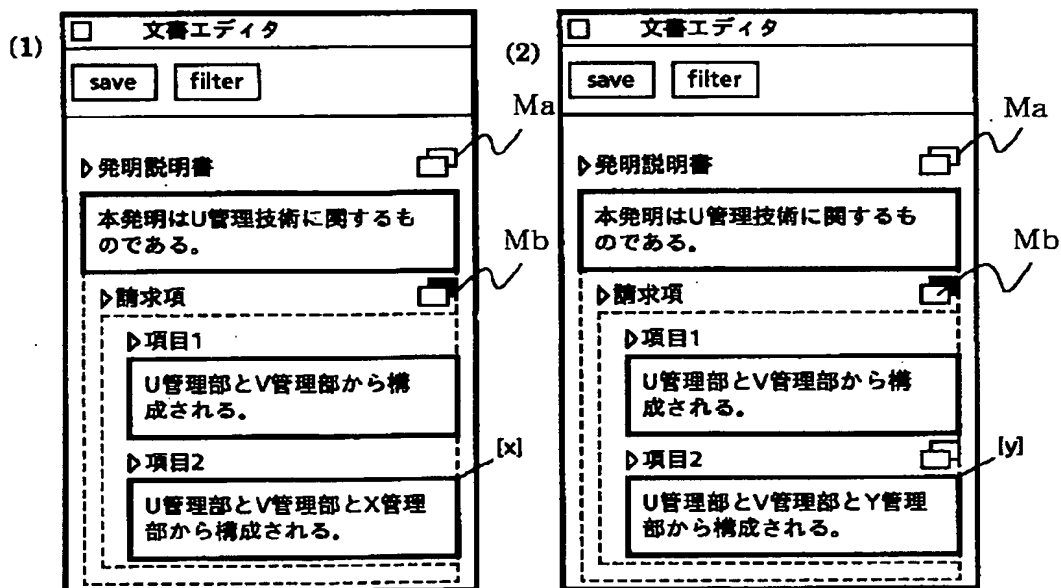
複合文書構造



【図37】

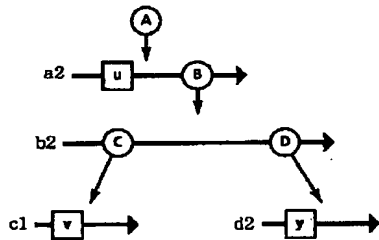


【図28】



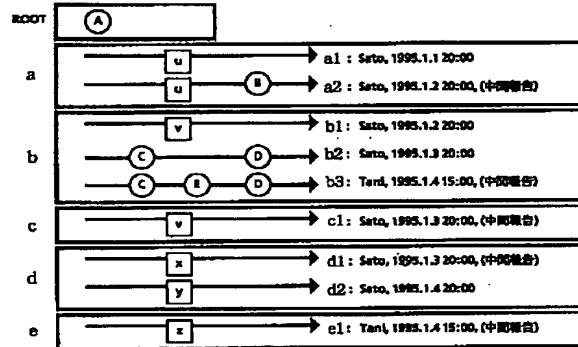
【図29】

文書構造



【図30】

複合文書構造



【図31】

(1) 文書エディタ起動

抽出条件設定ツール

文書ルート: 発明説明書

ok

基準バージョン: 中間報告

基準日時:

ユーザ指定: % 本人

(2) 抽出条件変更

抽出条件設定ツール

文書ルート: 発明説明書

ok

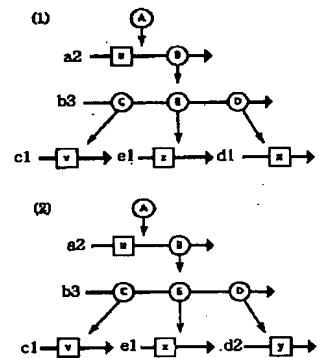
基準バージョン: 中間報告

基準日時:

ユーザ指定:

【図33】

文書構造



【図32】

(1)

文書エディタ

save filter

発明説明書

本発明はU管理技術に関するものである。

請求項

項目1

U管理部とV管理部から構成される。

項目1.5

U管理部とV管理部とZ管理部から構成される。

項目2

U管理部とV管理部とX管理部から構成される。

(2)

文書エディタ

save filter

発明説明書

本発明はU管理技術に関するものである。

請求項

項目1

U管理部とV管理部から構成される。

項目1.5

U管理部とV管理部とZ管理部から構成される。

項目2

U管理部とV管理部とY管理部から構成される。

Ma

Mb

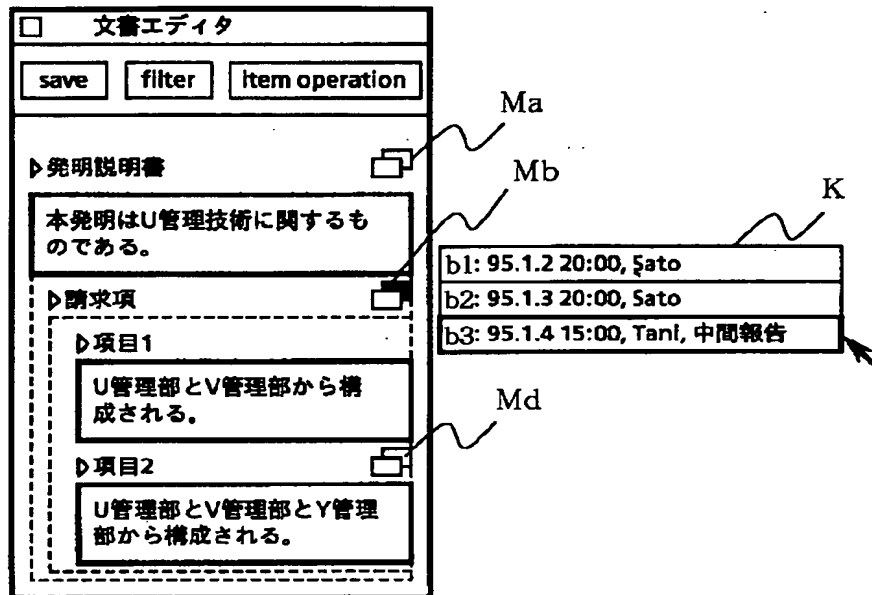
Md

Ma

Mb

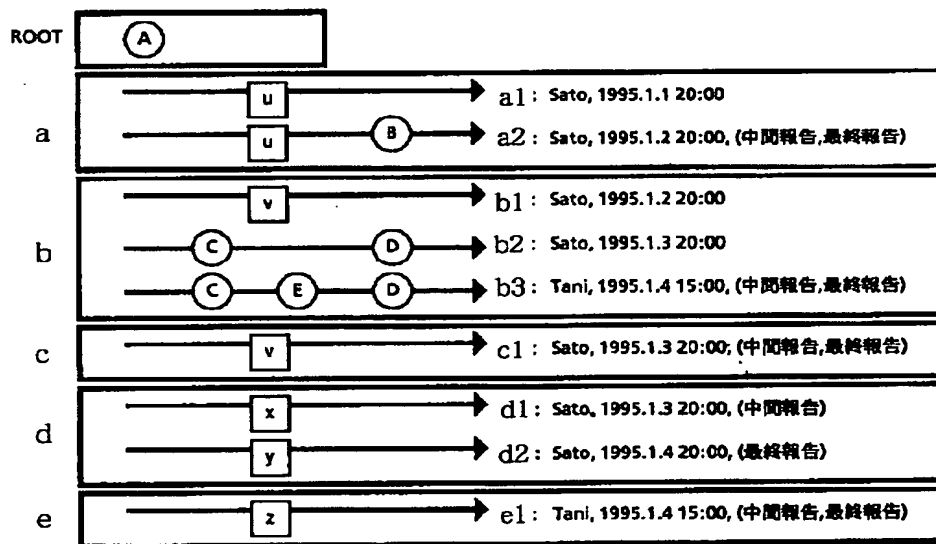
Md

【図34】

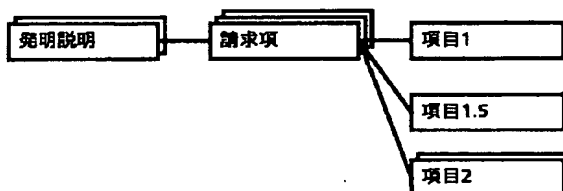


【図36】

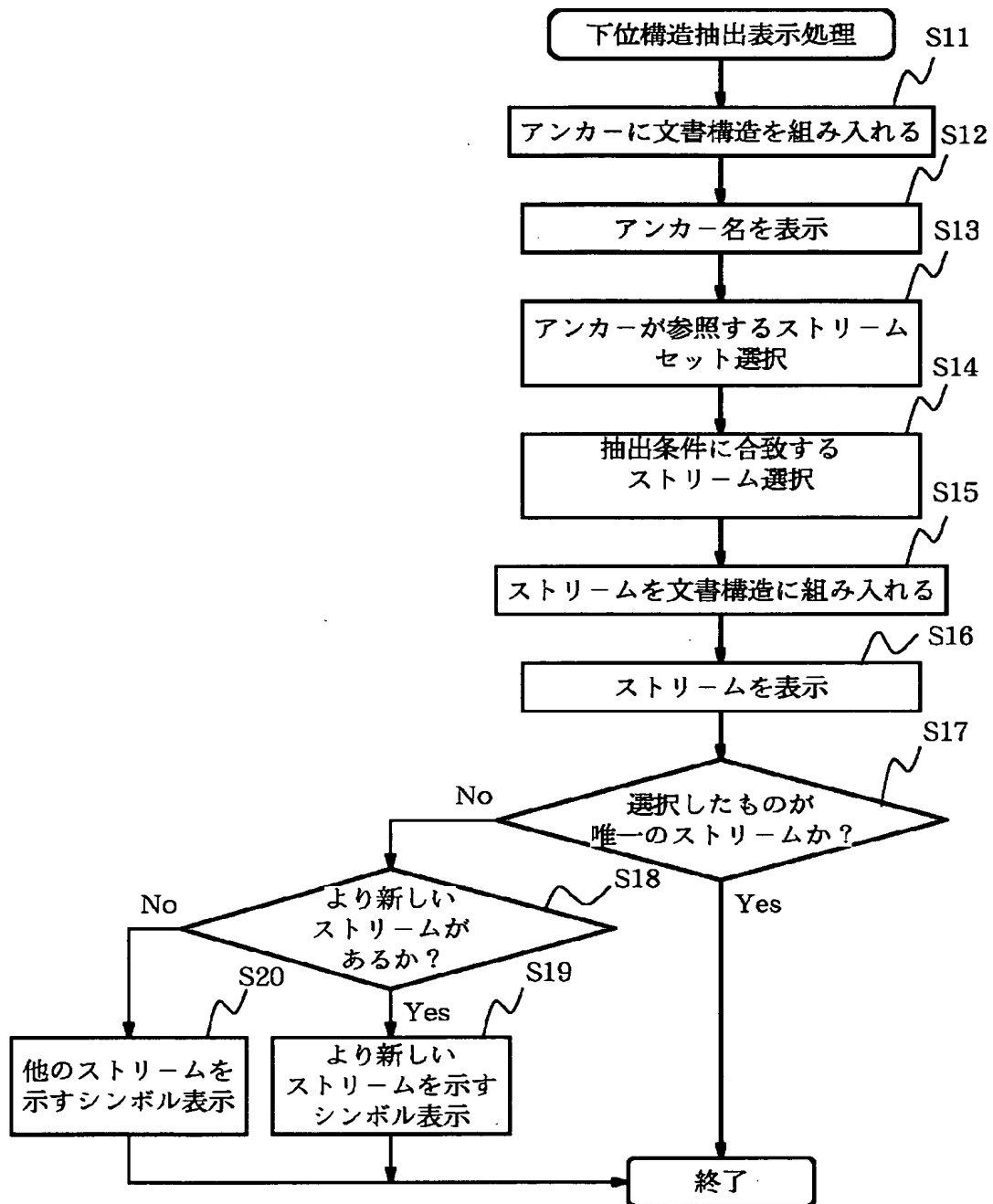
複合文書構造



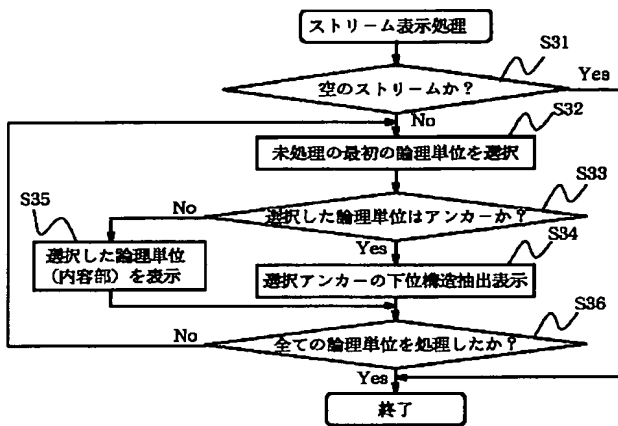
【図44】



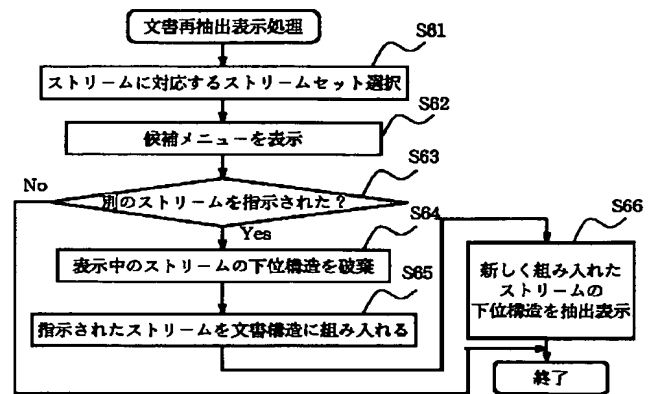
【図 38】



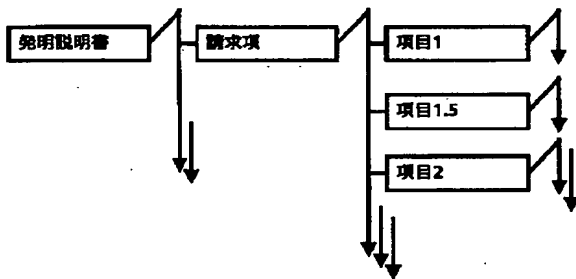
【図39】



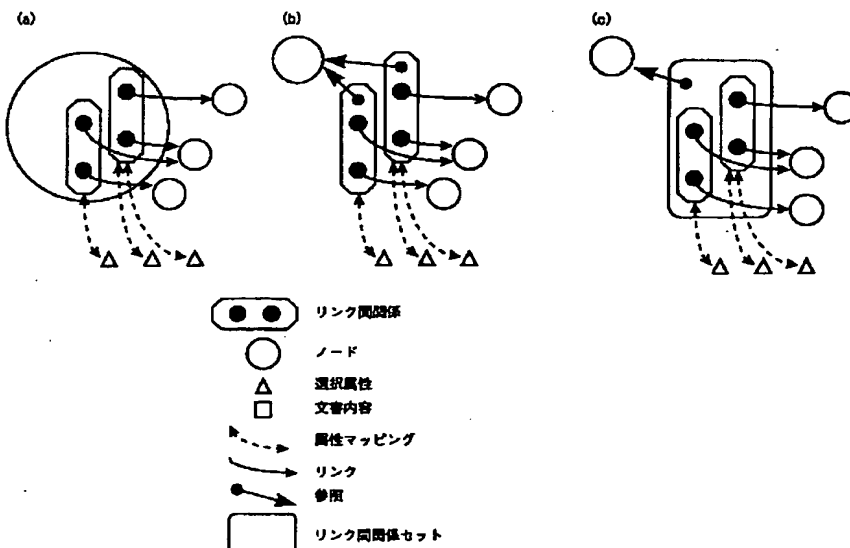
【図41】



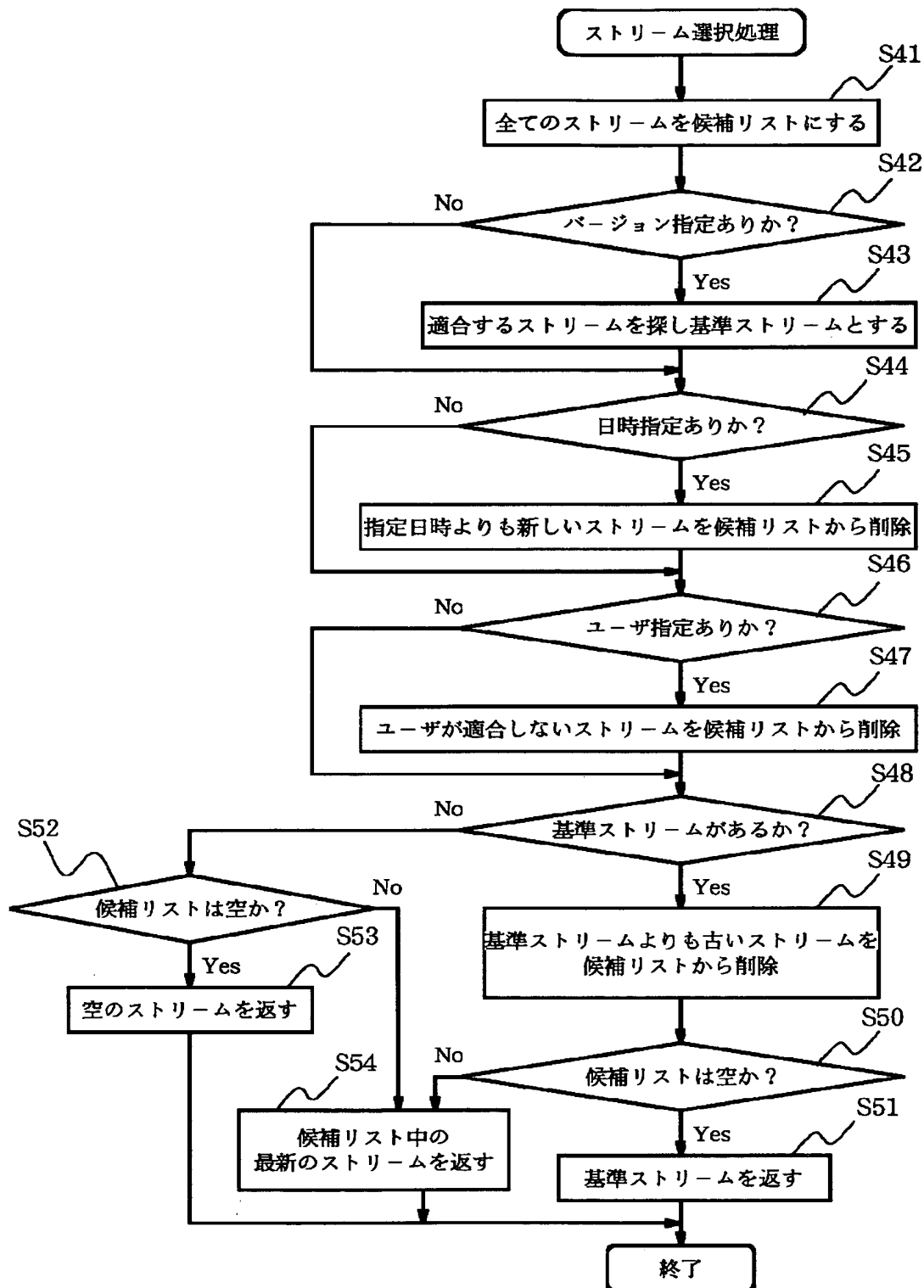
【図43】



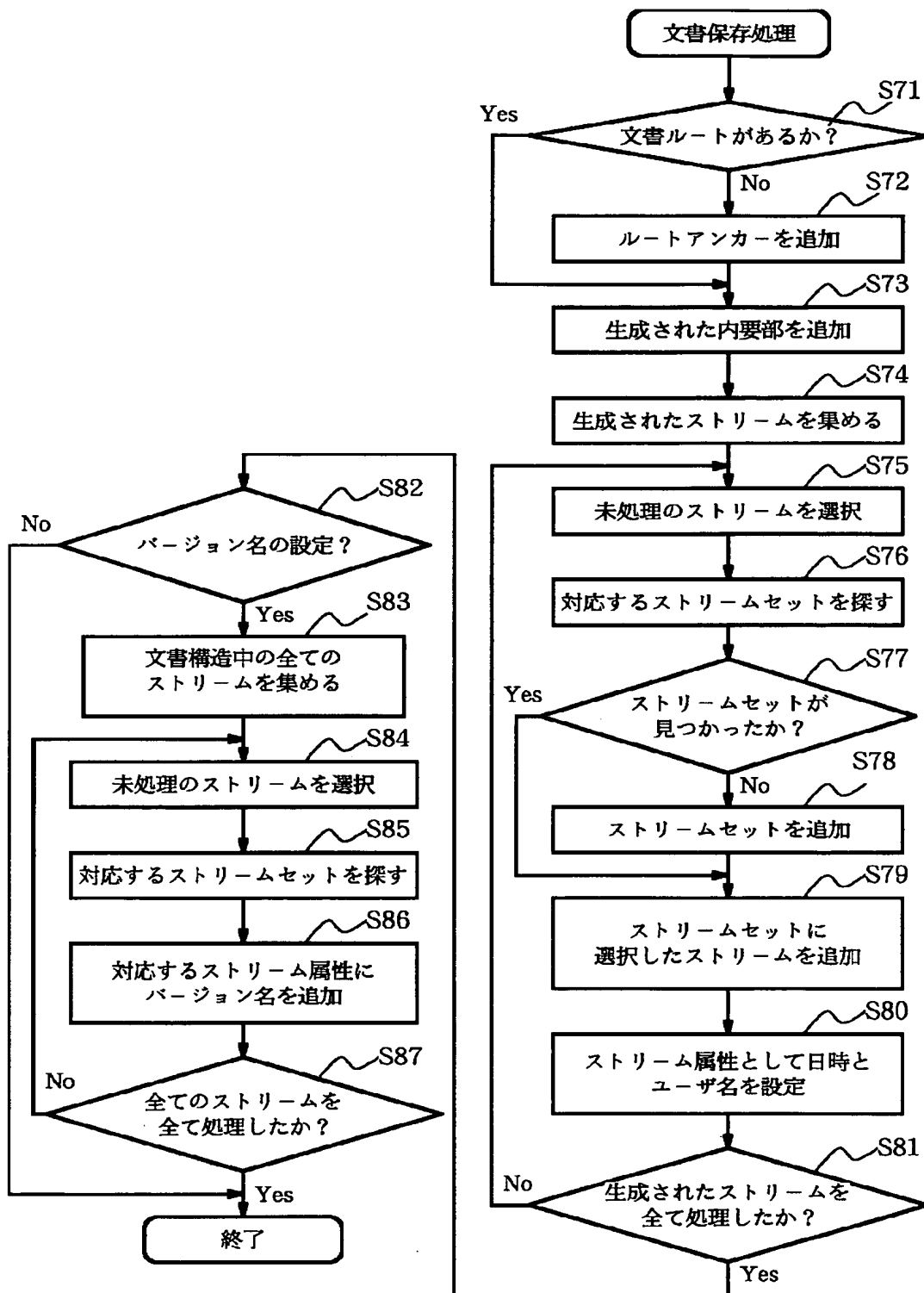
【図45】



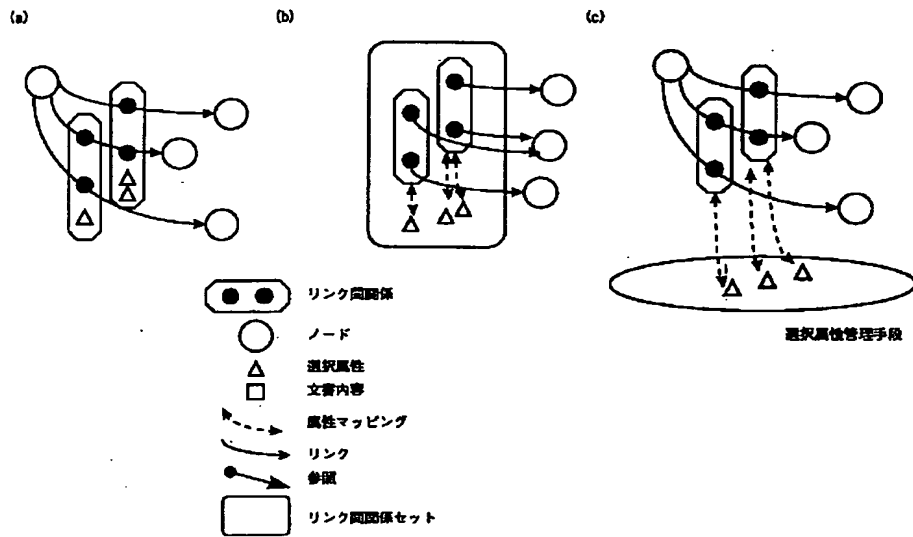
【図40】



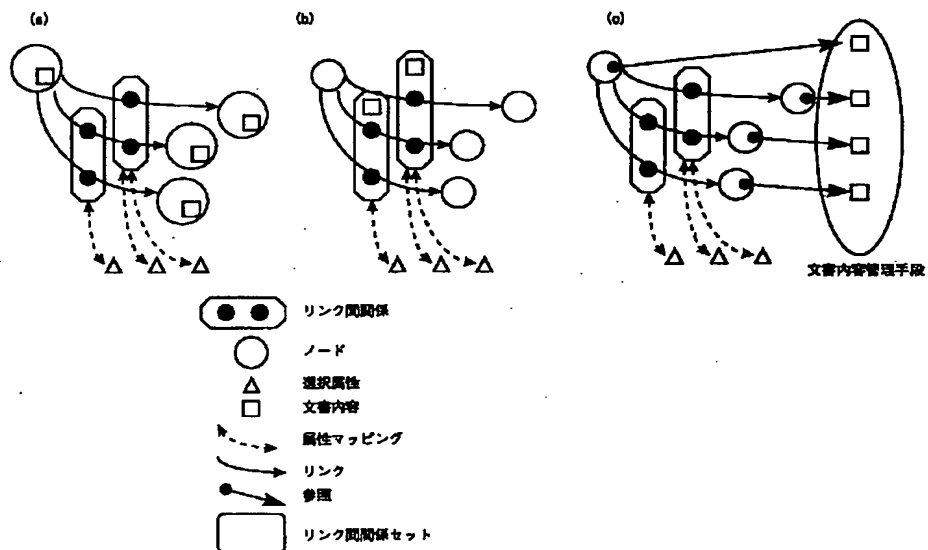
【図 42】



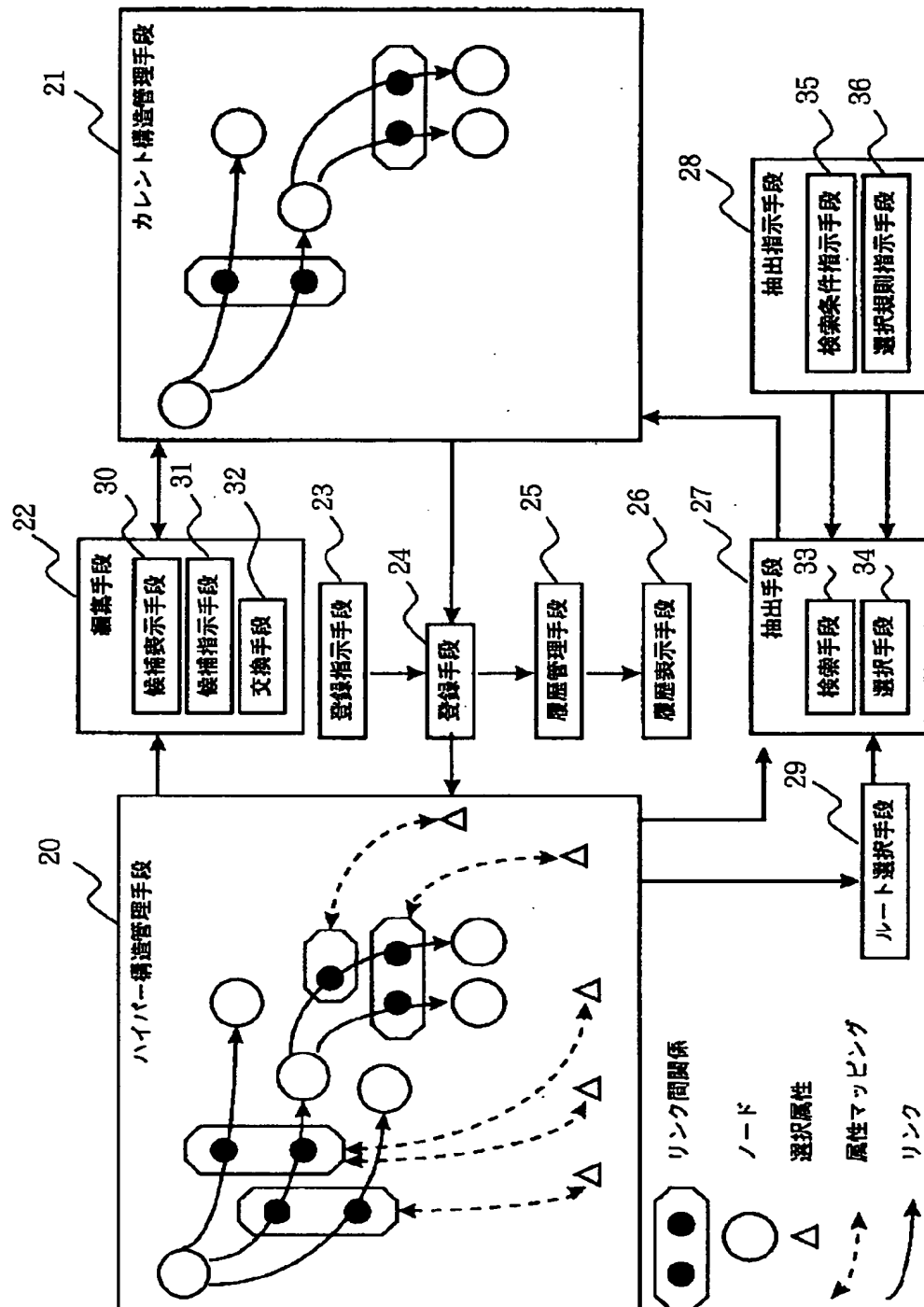
【図46】



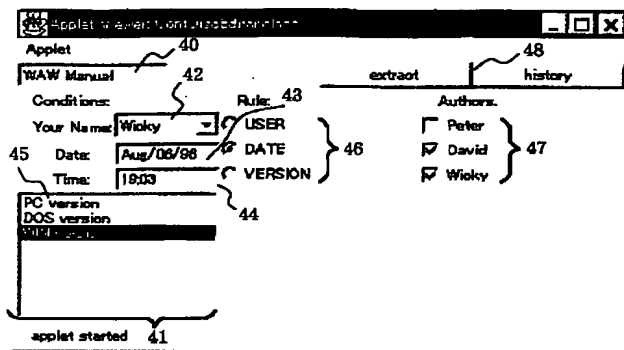
【図47】



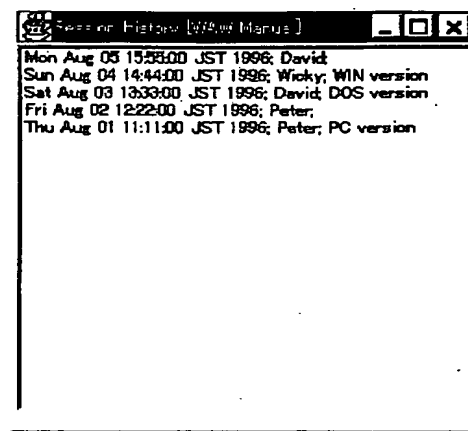
【図48】



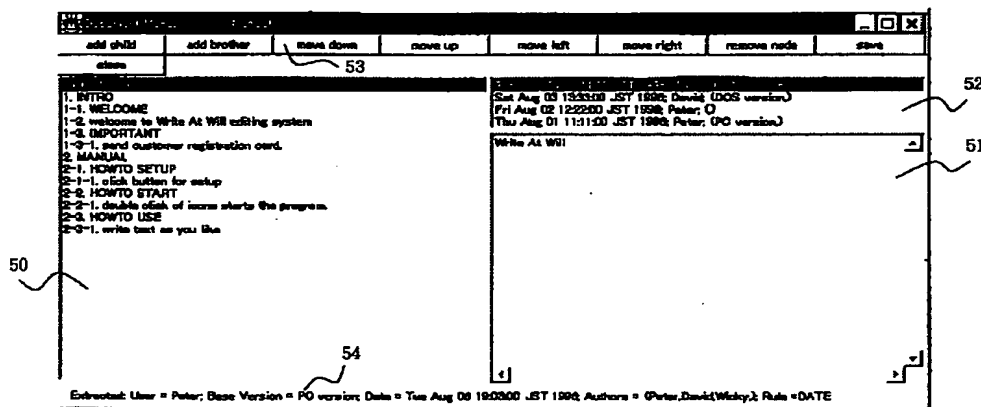
【図49】



【図50】

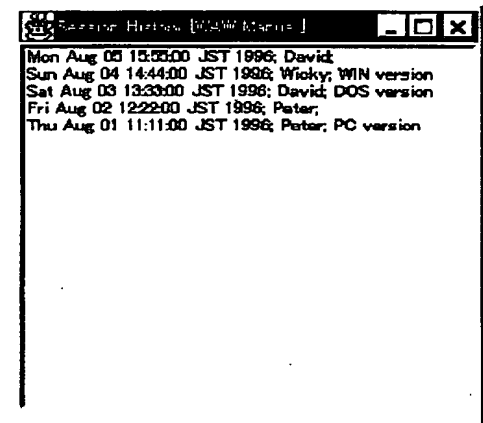
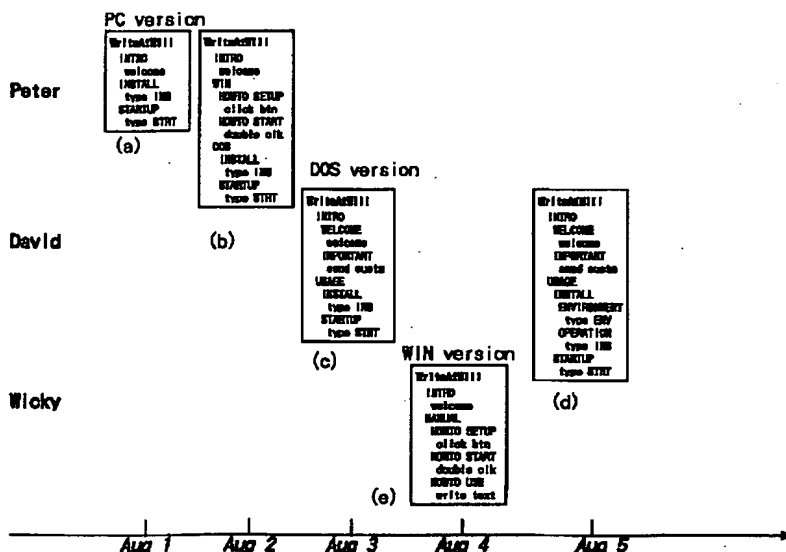


【図51】



【図52】

【図59】



Peter

```

By:task0011
INTRO
welcome
WIKI
NOTED SETUP
cd /etc /bin
NOTED START
double click
DOS
INSTALL
type /etc
STARTUP
type WIKI
  
```

(a)

David

```

By:task0011
INTRO
WELCOME
welcome
IMPORTANT
good advice
DAME
INSTALL
type /etc
STARTUP
type BUNT
  
```

(b)

(c)

Wicky

```

By:task0011
INTRO
welcome
SARIAL
NOTED SETUP
cd /etc /bin
NOTED START
double click
NOTED /etc
write text
  
```

(d)

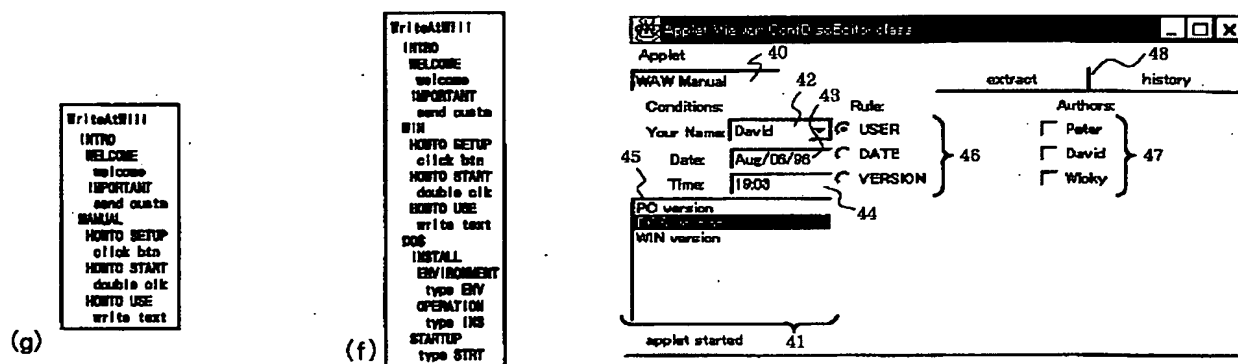
Aug 1 Aug 2 Aug 3 Aug 4 Aug 5

(a) Peter Aug 1

(b) Peter Aug 2

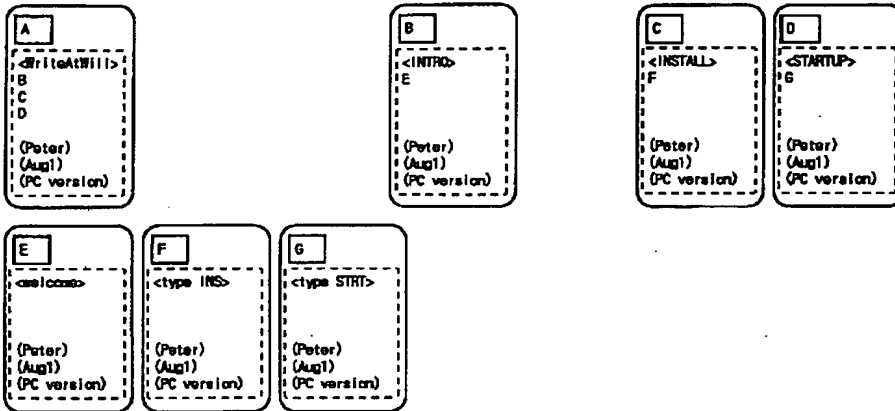
(c) David Aug 3

【図 60】



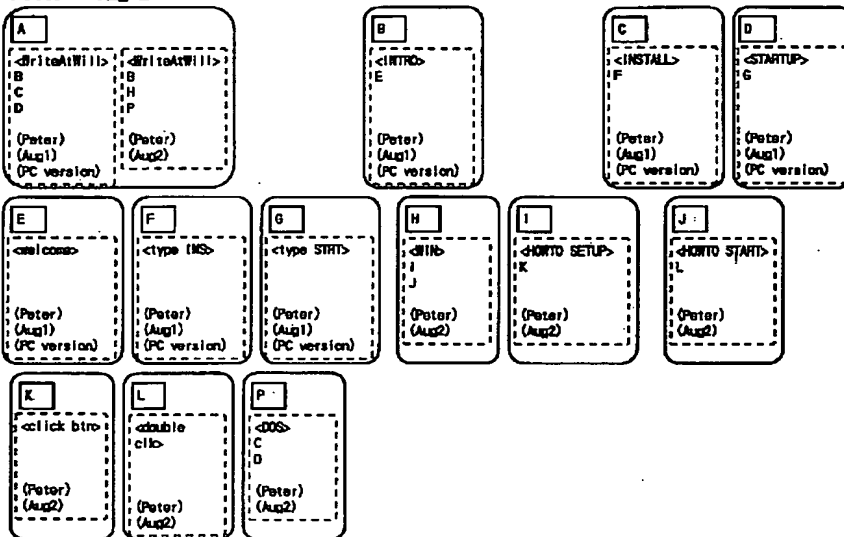
【図55】

Peter Aug 1



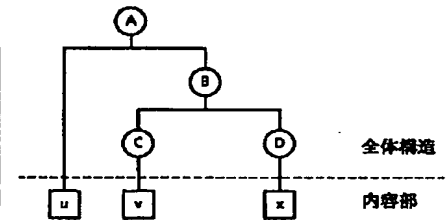
【図56】

Peter Aug 2

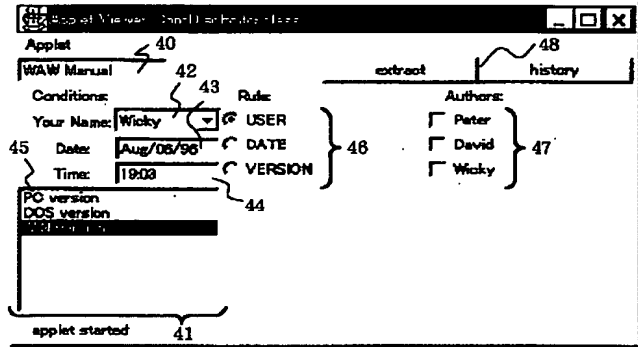
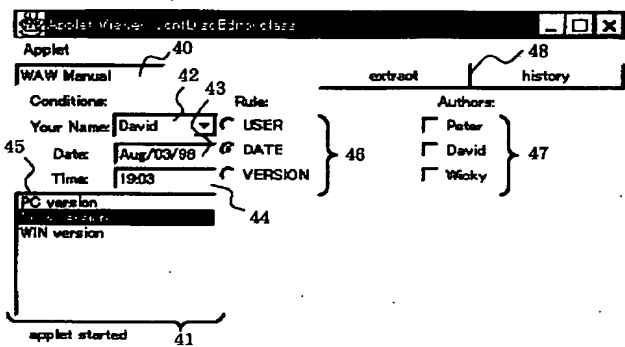


【図62】

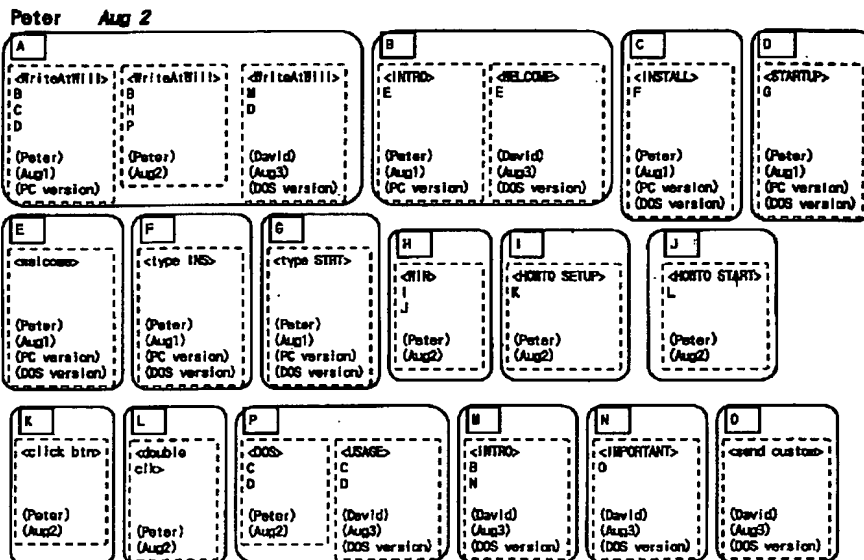
【図84】



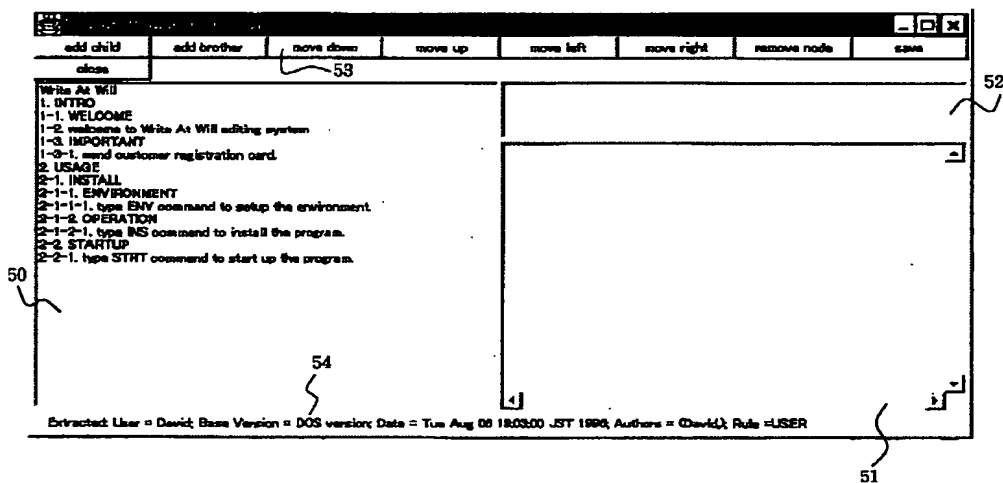
【図64】



【図57】

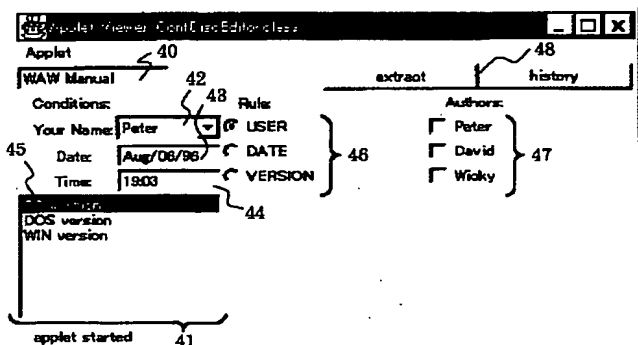
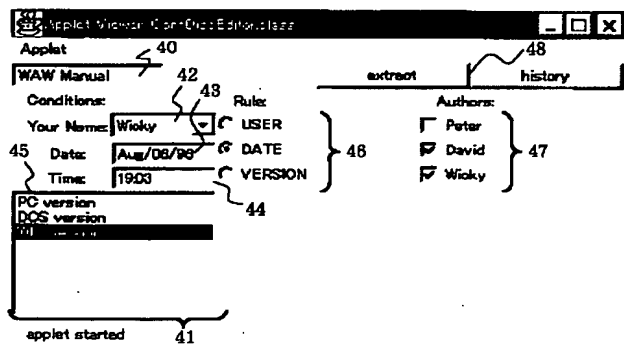


【図61】

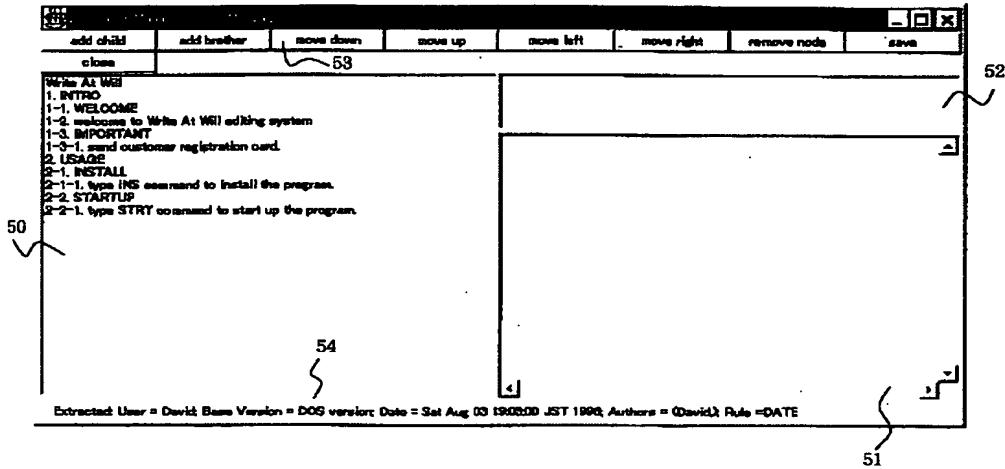


【図66】

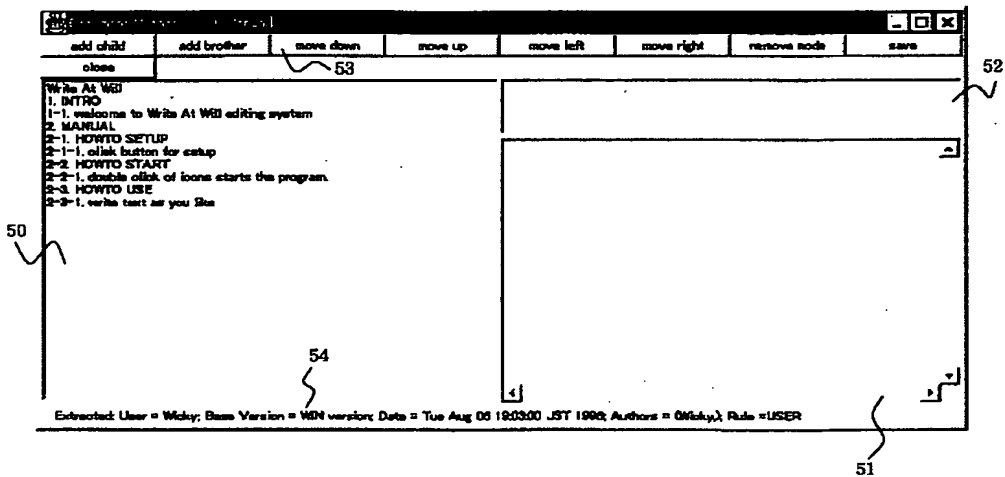
【図68】



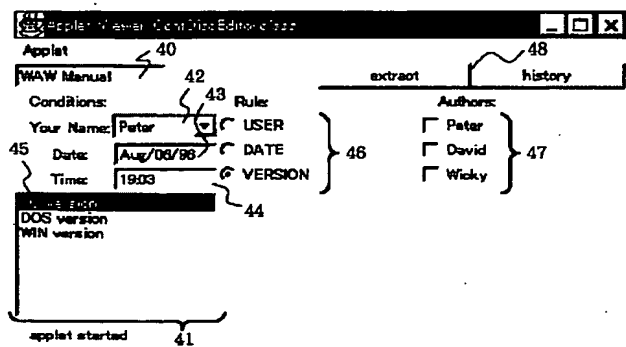
【図 6 3】



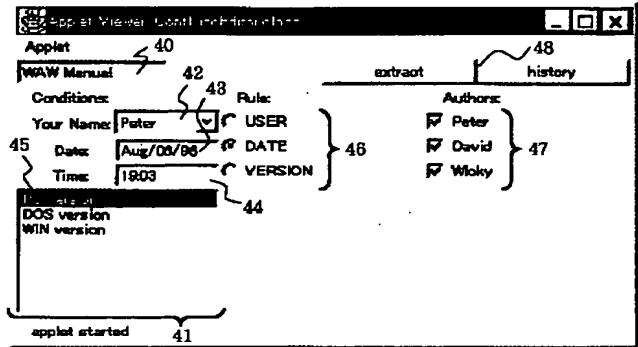
【図 6 5】



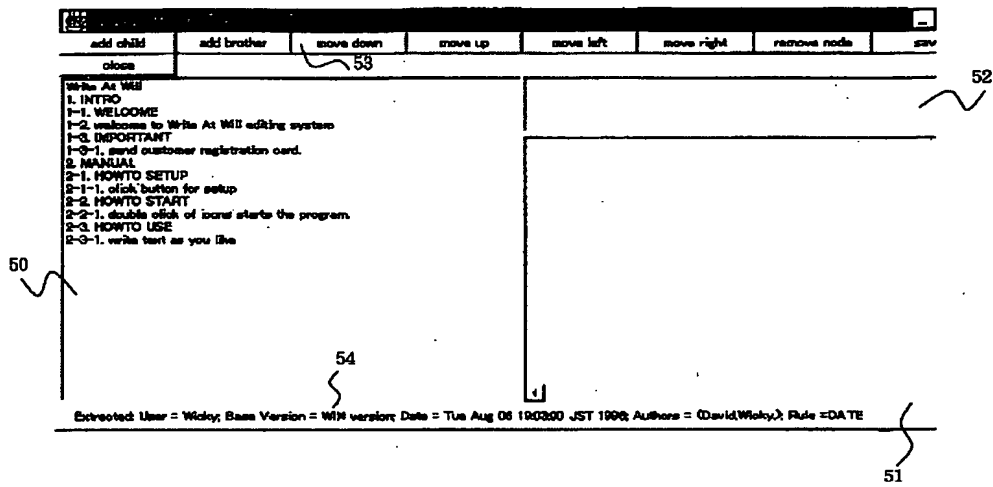
【図 7 0】



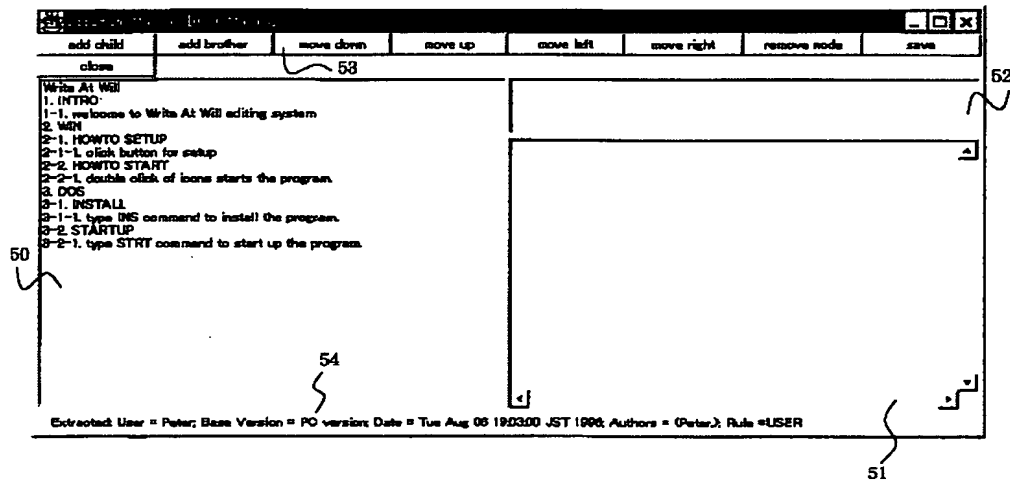
【図 7 2】



【図67】

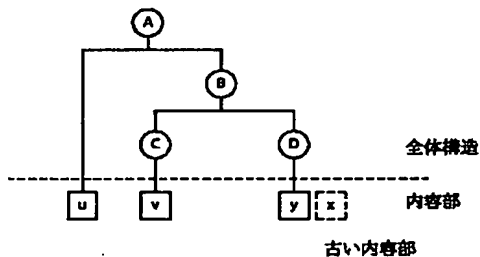


【図69】



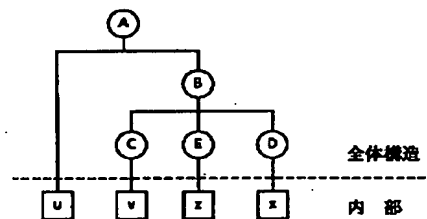
【図85】

内容部を変更する (x → y)

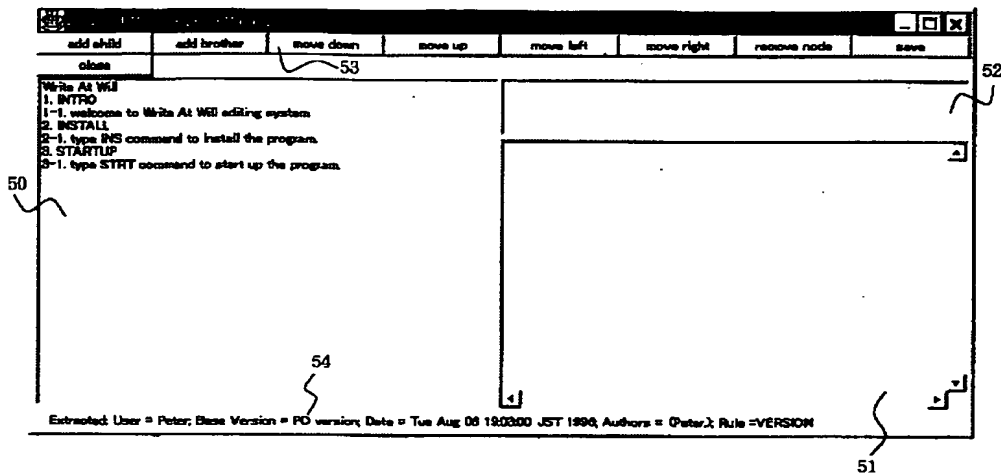


【図86】

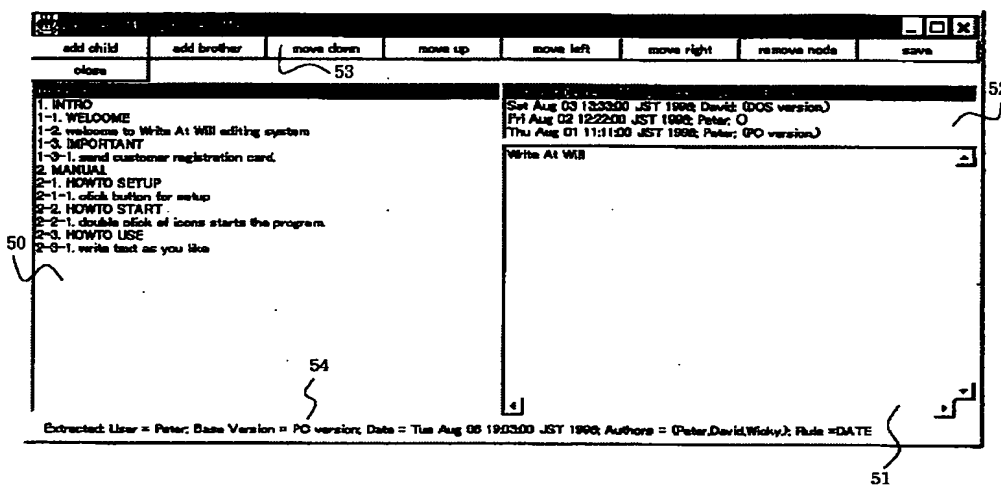
部分構造(E)を挿入する



【図71】

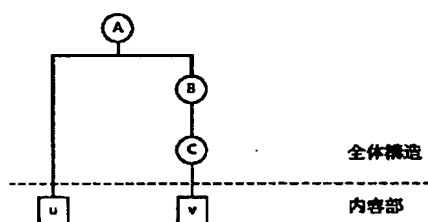


【図73】



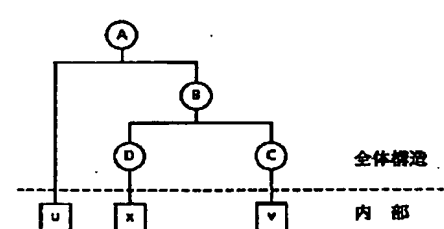
【図87】

部分構造(D)を削除する

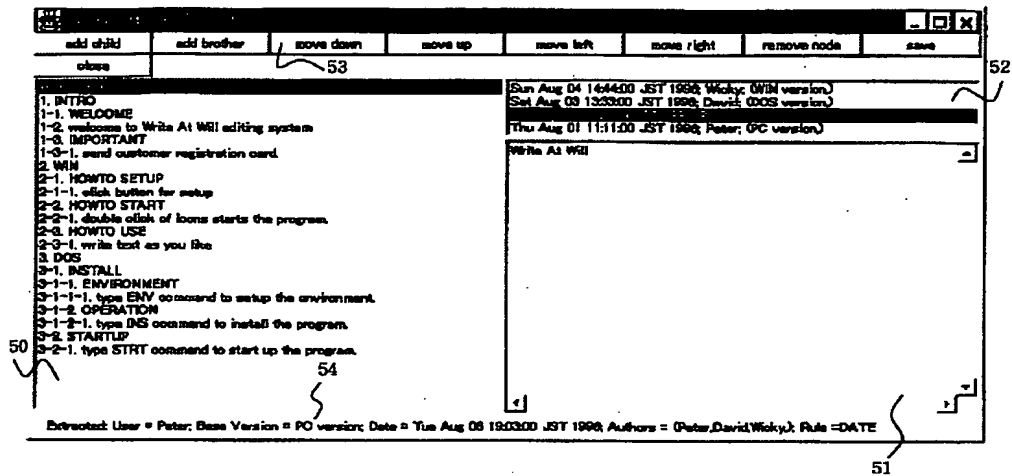


【図88】

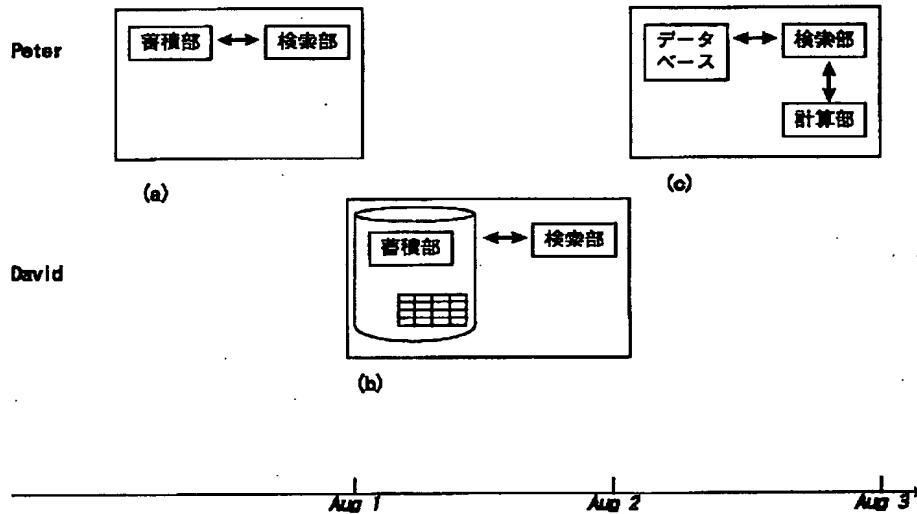
部分構造の順序を入れ替える(C↔D)



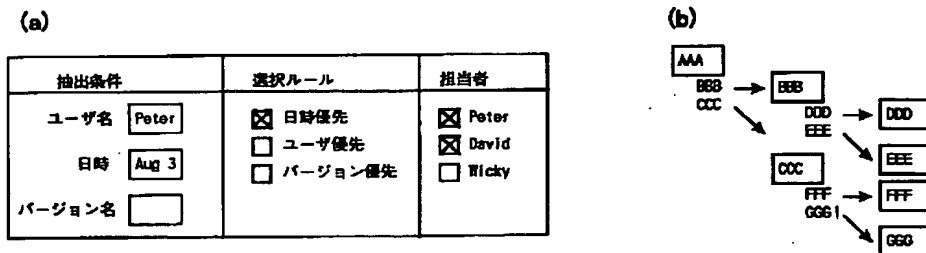
【図74】



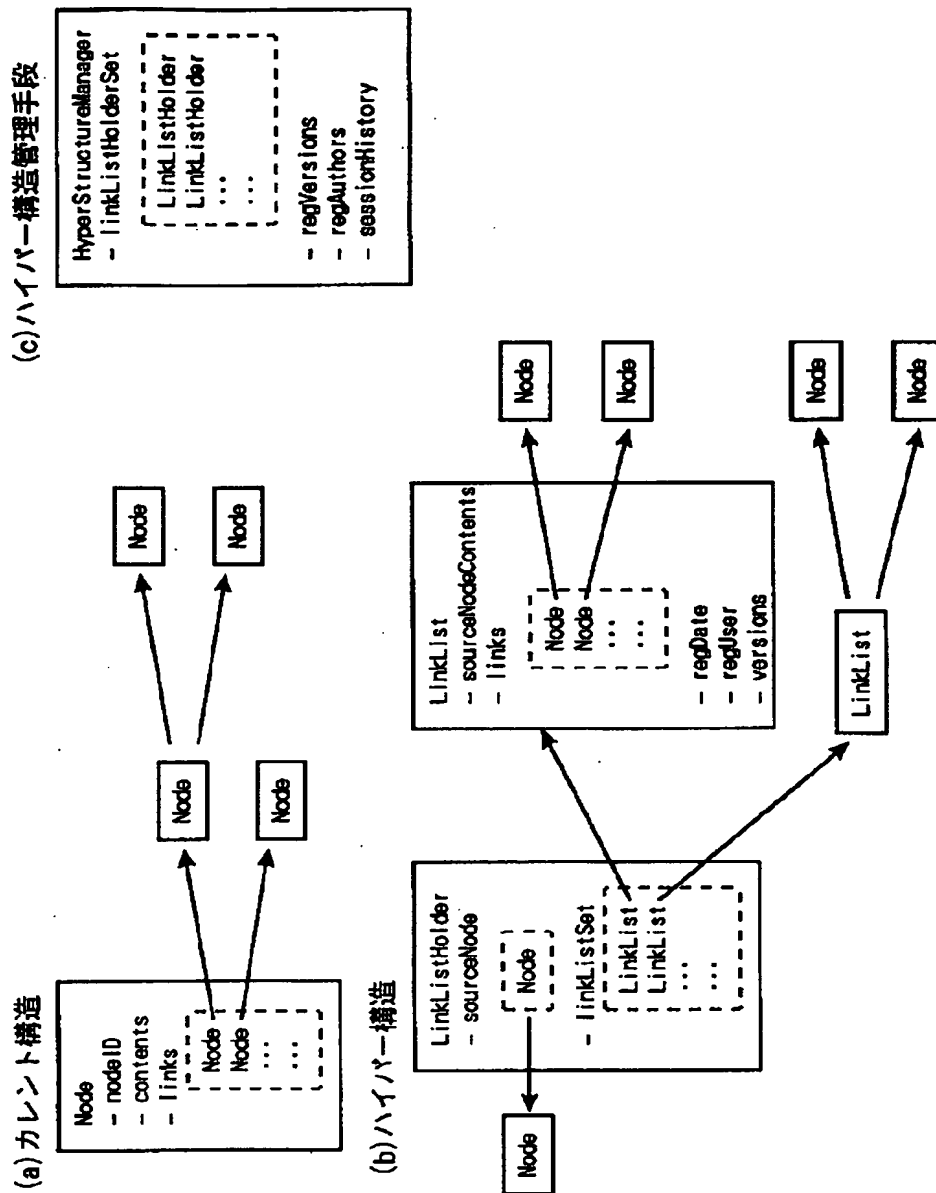
【図76】



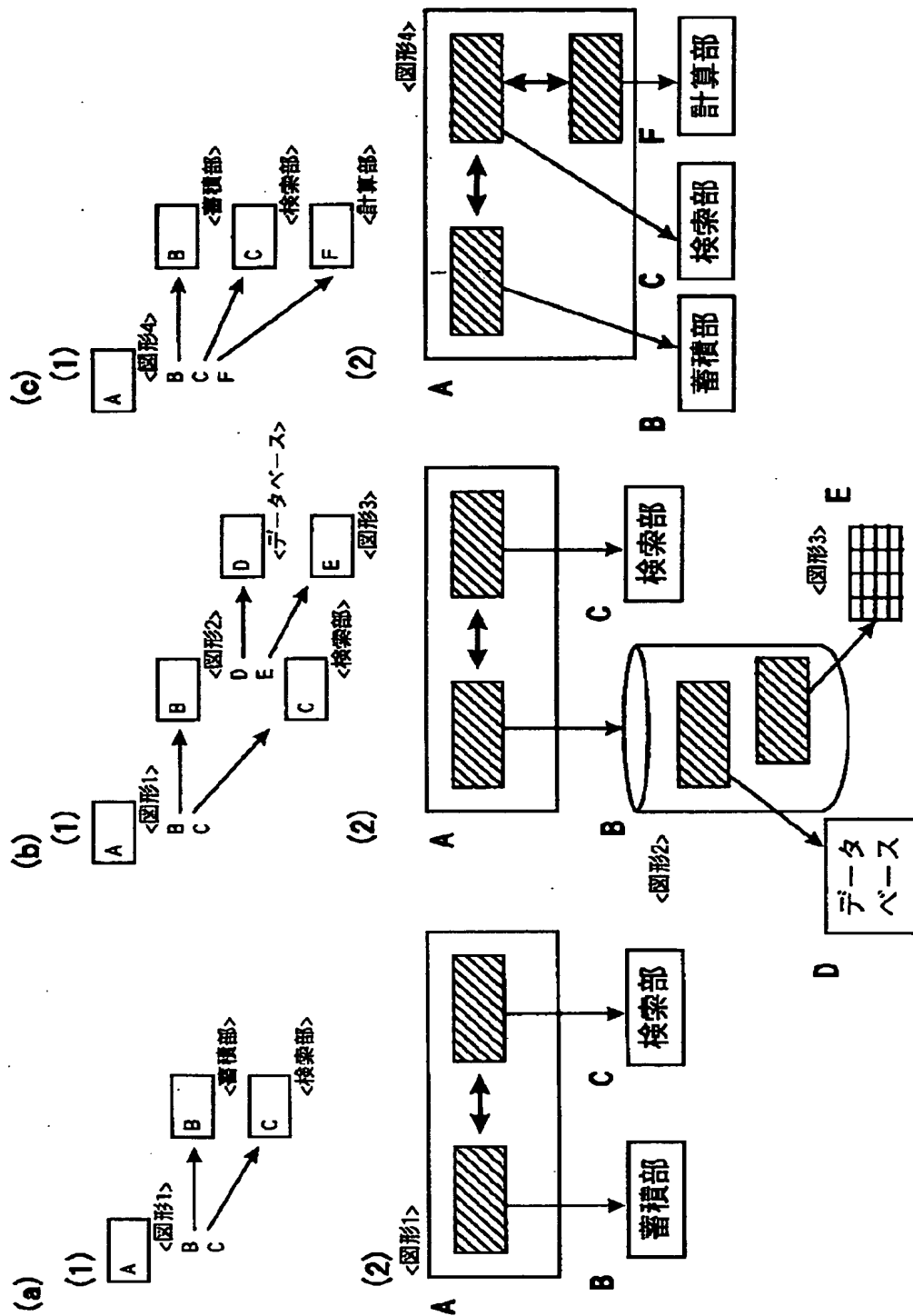
【図83】



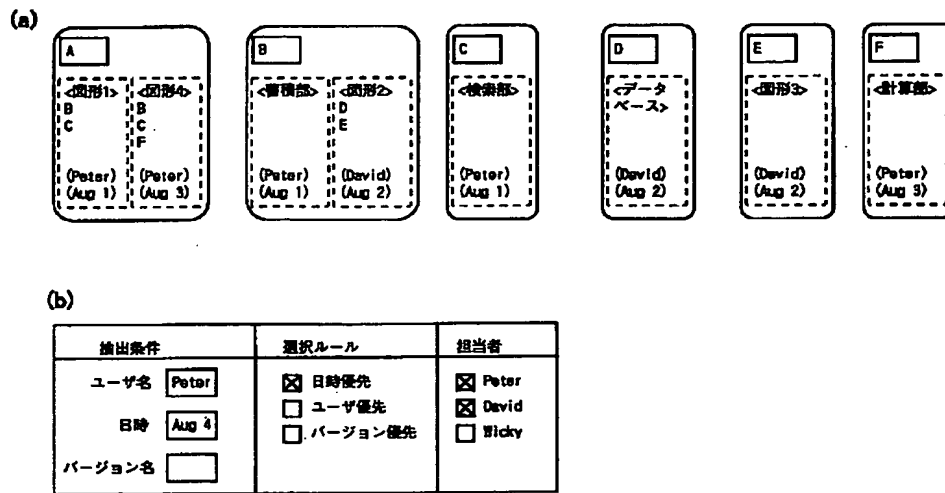
【図 7 5】



【図77】

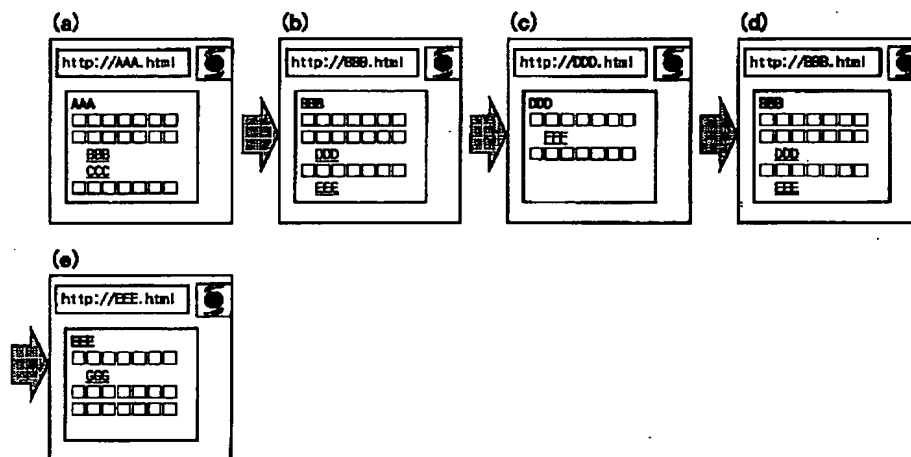


【図78】



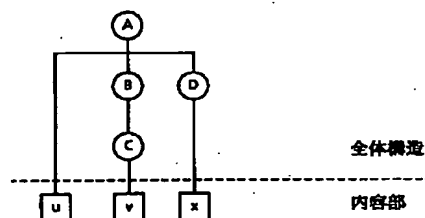
【図80】

Peter Aug 1

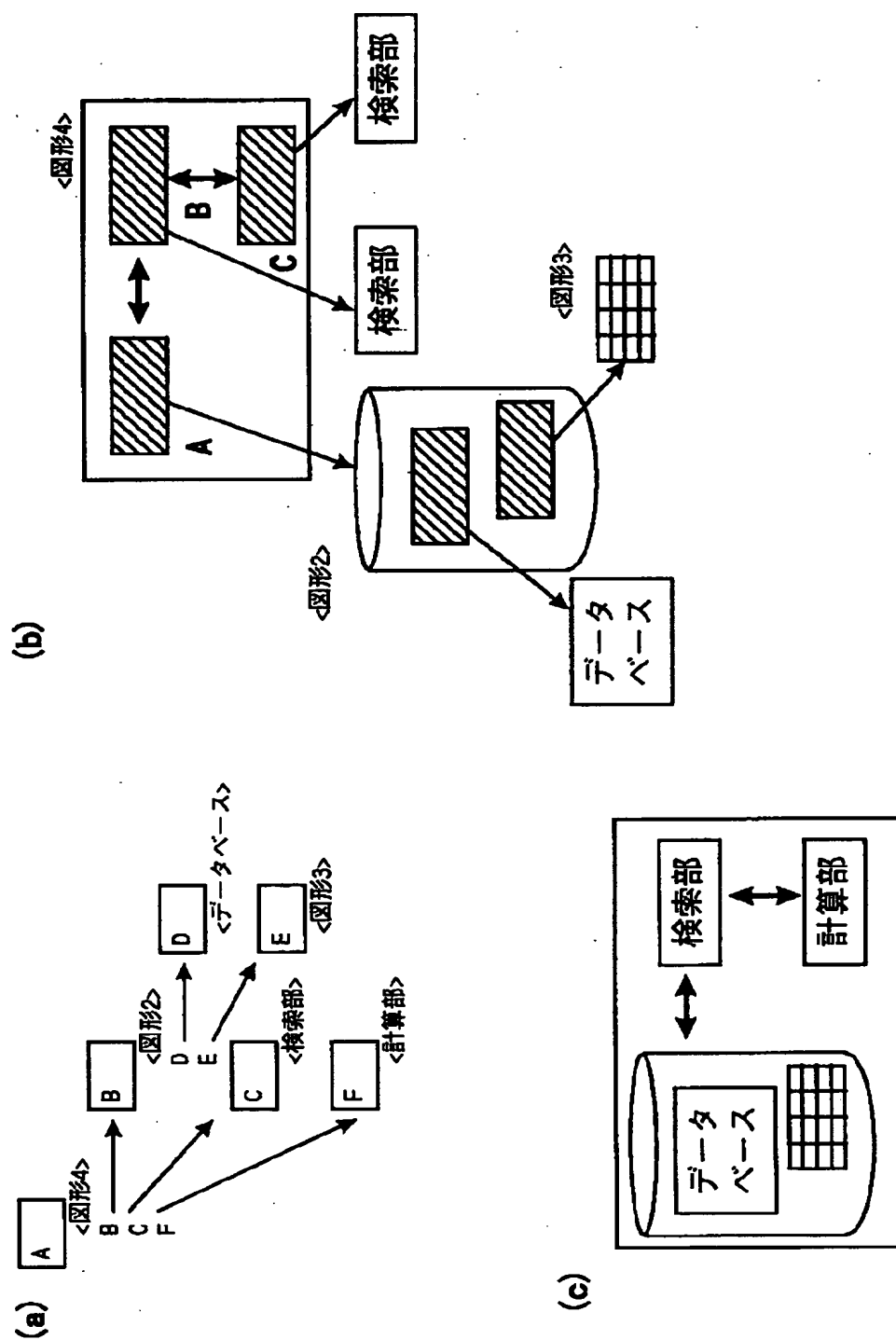


【図89】

部分構造(D)を移動する

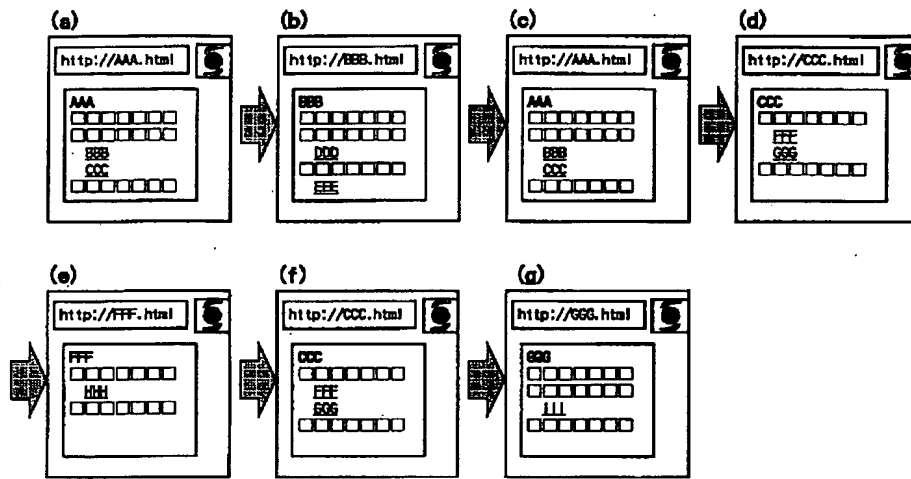


【図 7 9】



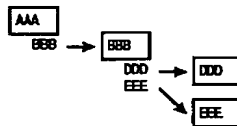
【図 8 1】

David Aug 2

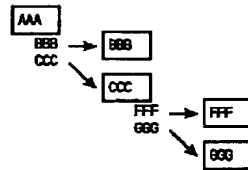


【図 8 2】

(a) Peter Aug 1



(b) David Aug 2



(a)

